LOCALIZED SPOT PATTERNS FOR THE BRUSSELATOR REACTION-DIFFUSION SYSTEM

A Thesis Submitted to the
College of Graduate Studies and Research
in Partial Fulfillment of the Requirements
for the degree of Master of Science
in the Department of Mathematics and Statistics
University of Saskatchewan
Saskatoon

by

Raed Ali Mara'Beh

Permission to Use

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Mathematics and Statistics
142 McLean Hall
106 Wiggins Road
University of Saskatchewan
Saskatoon, Saskatchewan
Canada
S7N 5C9

Abstract

The Brusselator reaction-diffusion model characterizes dynamical processes of some reaction diffusion systems in chemistry, physics, biology, and geology. On the sphere, the solutions of the Brusselator system center on a discrete set of points. In this thesis, we study the system of differential-algebraic equations (DAEs) that describes the slow dynamics of localized spot patterns for the Brusselator model on the surface of a unit sphere. The DAE system is solved numerically using Matlab's `ode15s` function.

The relationship between the equilibria of the DAE system and the set of elliptic Fekete points is studied. Precisely, solutions of DAE system are obtained from solving the elliptic Fekete optimization problem. The optimization problem is solved using the particle swarm optimization method. It is verified that for $N = 2, 3, \ldots, 8$ spots, the equilibrium spot configurations of the DAE system starting from a set of random initial points are elliptic Fekete points.

Acknowledgements

I would like to express my deep thanks to my supervisor Professor Raymond J. Spiteri for his unlimited support, instructions, guidance, and patience.

I wish to thank the faculty in the University of Saskatchewan Department of Mathematics and Statistics, and the college of Graduate Studies for providing me with a sufficient research environment, and financial assistance. I wish to thank my colleagues in the Numerical Simulation Laboratory for their support during the research of this thesis.

I am also thankful to many friends in this city who have helped me through these difficult years, I deeply appreciate their friendship.

My sincere thanks goes to my sweet family: my father Ali, my mother Aida, my brothers Mohammad, Saed, Baha', and Ala', without their love and encouragement I would not have finished this thesis.

My parents and brothers:

Distance means so little, because you mean so much.

Contents

List of Tables

List of Figures

List of Abbreviations

DAE	Differential-algebraic equation
PSO	Particle swarm optimization
RD	Reaction–diffusion

Chapter 1

Introduction

1.1 An overview

There is a wide range of body types within animals. Some animals, such as cheetahs and leopards, have spotted coats, whereas others are striped, like zebras. Depending upon the animal, the markings may be large or small. Some animals may have spotted bodies and striped tails, or vice versa. Some animals, such as elephants and mice, do not have coat patterns at all; see Figure 1.1. Generally, biology can provide a genetic explanation of the coat patterns, but the mechanism can be described by mathematics, precisely reaction-diffusion systems.

Figure 1.1: Coat pattern in some animals

Reaction–diffusion (RD) systems are mathematical models that describe how the con-

centration of one or more materials distributed in space change under the effect of two processes: chemical reaction and diffusion. In the case of animal coat patterns formation, a RD model describes the way in which two different chemical products react and diffuse on the skin, one which stimulates the production of melanin (colouring the skin), and one which inhibits this production. RD systems also arise widely in chemistry, chemical engineering, biology, geology, physics, ecology, and many other areas, [27, 28, 30, 40, 51].

Over the past decade, many references cover the RD patterns on the sphere and other compact manifolds, [8, 10, 21, 31, 33, 36, 52]. In the singularly perturbed limit corresponding to a large diffusivity ratio between two components, many RD systems admit localized spot patterns, where the solution concentrates at discrete points. This thesis studies localized spot patterns on the surface of the unit sphere for the Brusselator RD model. The theoretical and numerical frameworks for analyzing the equilibrium spot patterns for Brusselator model on a sphere were covered in [39, 48].

In [39], formal asymptotic methods are used to derive a nonlinear algebraic system that characterizes the dynamics of spots. The stability of spot patterns for the Brusselator model was also discussed. It was also shown that, in some cases, the localized spot patterns can provide some instabilities: self-replication, competition, or oscillatory instabilities.

In [48], a system of differential-algebraic equations (DAEs) that characterizes the slow dynamics of localized spot patterns for the Brusselator RD model on the sphere was constructed. Numerical and asymptotic solutions of the DAE system were also presented. Precisely, the numerical analysis was performed of the DAE system for $N = 2, 3, \ldots, 8$ spots. A fundamental connection between equilibria of the DAE system and a set of the elliptic Fekete points has been discussed recently in [39, 48].

1.2 Summary of Results

In this work, asymptotic and numerical analysis of the Brusselator RD model on a sphere are discussed. The DAE system is solved numerically, in the absence of the instabilities, using Matlab's `ode15s` function [42]. We obtain the equilibria of the DAE system from solving the classical optimization elliptic Fekete problem, and the `pyswarm` python package [1] is used to

solve the optimization problem. It is shown that a set of steady-state solutions of the DAE system for $N = 2, 3, \ldots, 8$ spots corresponds to the elliptic Fekete points set. The patterns of spot locations and strengths are presented for $N = 2, 3, \ldots, 12$ spots.

1.3 Thesis Outline

The outline of this thesis is as follows.

- Chapter 2 discusses the theory of the DAEs, some basic types of DAEs, and some methods for the numerical solutions of DAEs.
- Chapter 3 presents the patterns of Brusselator RD Model on the surface of a unit sphere, two principal results for slow spot dynamics, and the numerical simulations for $N = 6$ spots.
- Chapter 4 provides an introduction of the optimization problems, some categories of optimization problems, the particle swarm optimization algorithm (PSO), advantages of PSO, and some variants of PSO.
- Chapter 5 contains the results of the analysis, specifically, the minimal values of the logarithmic energy, the optimal locations of Fekete problem, the localized patterns for $N = 2, 3, \ldots, 12$ spots, and further relationships between Fekete points and the DAE equilibria.
- Chapter 6 contains a summary of the contributions and proposed future work.

Chapter 2

Differential-Algebraic Equations

2.1 Introduction

DAEs can be described as a system of ordinary differential equations coupled with algebraic equations. Many applications generate a family of DAE models, such as mechanical systems, optimal control, molecular dynamics, chemical kinetics, and chemical process control [4] [6]. In this chapter, we review the definition of DAEs and discuss some special types. Some numerical methods for solving DAEs are presented briefly.

A system of differential equations that is of the form

$$\mathbf{F}(t, \mathbf{y}, \dot{\mathbf{y}}) = \mathbf{0}, \tag{2.1}$$

is called DAE if the Jacobian matrix $\frac{\partial \mathbf{F}}{\partial \dot{\mathbf{y}}}$ is singular.

Example 2.1.1. ***Simple pendulum in Cartesian coordinates*** *[4]*

The motion of the simple pendulum can be expressed in terms of the Cartesian coordinates (x_1, x_2) *of a point particle at the end of the rod. With a Lagrange multiplier, Newton's equations of motion give*

$$\begin{aligned} \ddot{x}_1 &= -\lambda x_1, \\ \ddot{x}_2 &= -\lambda x_2 - g, \end{aligned} \tag{2.2}$$

where g *is the force of gravity and* $\lambda(t)$ *is a Lagrange multiplier. The fact that the rod has a*

fixed length 1 gives the additional relation

$$x_1^2 + x_2^2 = 1. \tag{2.3}$$

This system can be written as a first-order DAEs as follows

$$\begin{aligned} \dot{x}_1 &= x_3, \\ \dot{x}_2 &= x_4, \\ \dot{x}_3 &= -\lambda x_1, \\ \dot{x}_4 &= -\lambda x_2 - g, \\ x_1^2 + x_2^2 &= 1. \end{aligned} \tag{2.4}$$

2.2 Index of a DAE

In DAE, the repeated differentiation of the constraints might reduce it to an ordinary differential equation (ODE). This is valid except when the problem is singular. The number of differentiations required is called the index of the DAE.

Example 2.2.1. *Let $q(t)$ be a given smooth function.*

- *The scalar equation*

$$y = q(t), \tag{2.5}$$

  is a (trivial) index-1 DAE because an ODE can be obtained with one differentiation.

- *The system*

$$\begin{aligned} y_1 &= q(t), \\ y_2 &= \dot{y}_1, \end{aligned} \tag{2.6}$$

  is an index-2 DAE because the constraint must be differentiated twice to get an ODE for y_2.

- *The system*

$$\begin{aligned} u &= q(t), \\ y &= \ddot{u}, \end{aligned} \tag{2.7}$$

is an index-3 DAE because the constraint must be differentiated three times to get an ODE for y.

2.3 Classification of DAEs

A DAE in the form

$$\mathbf{F}(t, \mathbf{y}, \dot{\mathbf{y}}) = \mathbf{0}, \tag{2.8}$$

is called fully implicit, whereas DAE in the form

$$\begin{aligned} \dot{\mathbf{y}} &= \mathbf{f}(t, \mathbf{y}, \mathbf{z}), \\ \mathbf{0} &= \mathbf{g}(t, \mathbf{y}, \mathbf{z}), \end{aligned} \tag{2.9}$$

is called semi-explicit. In this form, $\mathbf{y}$ is referred to as the differential variable, while $\mathbf{z}$ is referred to as the algebraic variable. This is a nice form because the variables are decoupled. In general, any DAE can be transformed to semi-explicit form with index increased by one.

2.4 Hessenberg forms

The general DAE system (2.1) can include problems that are not well-defined mathematically and cannot be discretized directly. Fortunately, most DAEs that arise in practice can be expressed as a combination of ODEs coupled with constraints. As such, differential and algebraic variables can be explicitly identified and treated appropriately, and the algebraic variables may be eliminated (in principle) using the same number of differentiations. These are called Hessenberg (pure) forms and are given below.

Hessenberg index-1

$$\begin{aligned} \dot{\mathbf{y}} &= \mathbf{f}(t, \mathbf{y}, \mathbf{z}), \\ \mathbf{0} &= \mathbf{g}(t, \mathbf{y}, \mathbf{z}), \end{aligned} \tag{2.10}$$

where $\frac{\partial \mathbf{g}}{\partial \mathbf{z}}$ is assumed to be nonsingular for all t. Indeed, this is a semi-explicit index-1 DAE. We can solve for $\mathbf{z}$ in the algebraic equation (using the implicit function theorem) in terms of $\mathbf{y}, t$ (like implicit ODEs).

Hessenberg index-2

$$\begin{aligned} \dot{\mathbf{y}} &= \mathbf{f}(t, \mathbf{y}, \mathbf{z}), \\ \mathbf{0} &= \mathbf{g}(t, \mathbf{y}), \end{aligned} \tag{2.11}$$

where $\frac{\partial \mathbf{g}}{\partial \mathbf{y}} \frac{\partial \mathbf{f}}{\partial \mathbf{z}}$ is assumed to be nonsingular for all t. This is called a pure index-2 DAE.

Hessenberg index-3

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(t, \mathbf{x}, \mathbf{y}, \mathbf{z}), \\ \dot{\mathbf{y}} &= \mathbf{g}(t, \mathbf{x}, \mathbf{y}), \\ \mathbf{0} &= \mathbf{h}(t, \mathbf{y}), \end{aligned} \tag{2.12}$$

provided $\frac{\partial \mathbf{h}}{\partial \mathbf{y}} \frac{\partial \mathbf{g}}{\partial \mathbf{x}} \frac{\partial \mathbf{f}}{\partial \mathbf{z}}$ is nonsingular for any t.

2.5 Numerical methods for DAEs

Numerical approaches for the solution of DAEs can be categorized into two classes:

i. Direct discretization

ii. Reformulation (index reduction) and discretization.

Direct discretization is a desirable method because a reformulation may be costly, it may require more input from the user, or it may involve more user intervention. However,

direct discretization is valid only for index-1 and semi-explicit index-2 DAEs, and so the reformulation method is popular.

Fortunately, many DAEs encountered in practical applications are either index 1 or, if higher-index, can be expressed as a simple combination of Hessenberg systems. However, some complications may arise, and the most robust direct applications of numerical ODE methods do not always work as one might hope, even for these restricted classes of problems. For a DAE of index greater than two, it is usually best to use one of the index-reduction techniques to convert the problem in a lower-index form.

2.5.1 Direct Discretization: Backward Euler

Consider the general DAE

$$\mathbf{F}(t, \mathbf{y}, \dot{\mathbf{y}}) = \mathbf{0}, \tag{2.13}$$

with $\mathbf{y} = (y_1, y_2, \ldots, y_m)$. Applying the backward Euler method to this DAE, we obtain

$$\mathbf{F}\left(t_n, \mathbf{y}_n, \frac{\mathbf{y}_n - \mathbf{y}_{n-1}}{\Delta t_n}\right) = \mathbf{0}. \tag{2.14}$$

This gives a system of m nonlinear equations for $\mathbf{y}_n$ at each t_n. Unfortunately, this simple method does not always work. In the worst case, there are simple higher-index DAE systems with well-defined solutions for which backward Euler (and in fact all other multi-step and Rung–Kutta methods) are unstable or not defined.

Consider the semi-explicit index-1 DAE

$$\begin{aligned} \dot{\mathbf{y}} &= \mathbf{F}(\mathbf{t}, \mathbf{y}, \mathbf{z}), \\ \mathbf{0} &= \mathbf{g}(t, \mathbf{y}, \mathbf{z}), \qquad \tfrac{\partial \mathbf{g}}{\partial \mathbf{z}} \text{ nonsingular .} \end{aligned} \tag{2.15}$$

Consistent with the implicit function theorem, there exists a function $\mathbf{G}$ such that

$$\mathbf{z} = \mathbf{G}(t, \mathbf{y}). \tag{2.16}$$

Therefore, the DAE system (2.15) is equivalent to the ODE

$$\dot{\mathbf{y}} = \mathbf{f}(t, \mathbf{y}, \mathbf{G}(t, \mathbf{y})). \tag{2.17}$$

Now, the backward Euler method can be applied to (2.15)

$$\begin{aligned} \tfrac{\mathbf{y_n} - \mathbf{y_{n-1}}}{\Delta t_n} &= \mathbf{f}(t_n, \mathbf{y_n}, \mathbf{z_n}), \\ \mathbf{0} &= \mathbf{g}(t_n, \mathbf{y_n}, \mathbf{z_n}). \end{aligned} \tag{2.18}$$

Solving for $\mathbf{z_n}$ and substituting into the algebraic equation yields

$$\frac{\mathbf{y_n} - \mathbf{x_{n-1}}}{\Delta t_n} = \mathbf{f}(t_n, \mathbf{x_n}, \mathbf{G}(t_n, \mathbf{y_n})), \tag{2.19}$$

which is the backward Euler discretization of the underlying ODE (2.17). Backward Euler is first-order accurate, stable, and convergent for semi-explicit index-1 DAEs. The same results can be shown for fully implicit index-1 DAEs and semi-explicit index-2 DAEs [4].

2.5.2 Backward differentiation formulas (BDF) and implicit Rung–Kutta methods

Backward differentiation formulas (BDF) can be applied with a constant stepsize Δt to a general nonlinear DAE to obtain

$$\mathbf{F}\left(t_n, \mathbf{y_n}, \frac{1}{\beta_0 \Delta t} \sum_{j=0}^{k} \alpha_j \mathbf{y_{n-j}}\right) = \mathbf{0}, \tag{2.20}$$

where β_0 and α_j, $j = 0, 1, \ldots, k$, are the coefficients of the BDF method. This method is the most widely used method for the numerical solution of DAEs, and most software packages

based on BDF methods to solve fully implicit index-1 DAEs. Fortunately, many problems from applications naturally arise in this form.

Implicit Runge–Kutta Methods

The s-stage implicit Runge–Kutta method applied to the general nonlinear DAE is given by

$$\begin{aligned} \mathbf{0} &= \mathbf{F}\left(t_{ni}, \mathbf{Y}_i, \mathbf{K}_i\right), \\ t_{ni} &= t_{n-1} + \Delta t c_i, \\ \mathbf{Y}_i &= \mathbf{y}_{n-1} + \Delta t \sum_{j=1}^{s} a_{ij}\mathbf{K}_j, \quad i = 1, 2, \ldots, s, \\ \mathbf{y}_n &= \mathbf{y}_{n-1} + \Delta t \sum_{i=1}^{s} b_i \mathbf{K}_i. \end{aligned} \tag{2.21}$$

We assume that the coefficient matrix $A = [a_{i,j}]$ is nonsingular.
For the semi-explicit problem, the internal stages satisfy

$$\begin{aligned} \mathbf{K}_i &= \mathbf{f}\left(t_{ni}, \mathbf{X}_i, \mathbf{Z}_i\right), \\ \mathbf{X}_i &= \mathbf{x}_{n-1} + \Delta t_n \sum_{j=1}^{s} a_{ij}\mathbf{K}_j, \\ \mathbf{0} &= \mathbf{g}(t_i, \mathbf{X}_i, \mathbf{Z}_i). \end{aligned} \tag{2.22}$$

For the algebraic variables $\mathbf{z}$, it is often better to avoid the quadrature step implied by the last step of a usual Runge–Kutta method because there is no corresponding integration in the DAE.

This gives an advantage to stiffly accurate methods which satisfy $b_j = a_{sj}$ because for these methods, the constraints are automatically satisfied at the final method stage; i.e.,

$$\mathbf{y}_n = \mathbf{Y}_n, \tag{2.23}$$

and there is no need for the final (quadrature) step.

Chapter 3

The Brusselator Model on the Surface of the Sphere

3.1 Introduction

The dimensionless standard form of the Brusselator [35] model formulated on the surface of the unit sphere can be given in terms of the activator $u = u(x,t)$ and the inhibitor $v = v(x,t)$ as

$$\begin{aligned} \frac{\partial u}{\partial t} &= \epsilon^2 \Delta_s u + \epsilon^2 E - u + f u^2 v, \\ \tau \frac{\partial v}{\partial t} &= \Delta_s v + \epsilon^{-2}(u - u^2 v), \end{aligned} \tag{3.1}$$

for some constants $E > 0$, $\tau > 0$, and $0 < f < 1$. Here the surface Laplacian Δ_s is the defined by

$$\Delta_s \equiv \frac{1}{\sin^2\phi}\frac{\partial^2}{\partial\theta^2} + \frac{1}{\sin\phi}\frac{\partial}{\partial\phi}\left[\sin(\phi)\frac{\partial}{\partial\phi}\right], \quad 0 < \phi < \pi, \quad 0 < \theta \leq 2\pi. \tag{3.2}$$

For $\epsilon \ll 1$, u can be localized in the domain as a spot pattern, i.e., a concentration at a discrete set of points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$.

The main goal of this chapter is to introduce the system of DAEs that describes the slow dynamics of spot patterns for the Brusselator model. In Section 3.4, the DAE system is solved numerically for $N = 6$ points.

3.2 Core problem

Consider a set of N spots with positions $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$. On the tangent plane to the sphere near the spot at $\mathbf{x} = \mathbf{x}_j$, the core problem is given on $0 < \rho < \infty$ as [39]:

$$\begin{aligned} &\Delta_\rho U_{j0} - U_{j0} + f U_{j0}^2 V_{j0} = 0, \qquad \Delta_\rho V_{j0} + U_{j0} - U_{j0}^2 V_{j0} = 0, \\ &U_{j0}'(0) = V_{j0}' = 0; \quad U_{j0} \to 0, \quad v_{j0} \sim S_j \log\rho + \chi + o(1) \ \text{ as } \rho \to \infty. \end{aligned} \tag{3.3}$$

where $U_{j0} = U_{j0}(\rho)$, $V_{j0} = V_{j0}(\rho)$ and $\Delta_\rho \equiv \frac{\partial^2}{\partial\rho^2} + \frac{1}{\rho}\frac{\partial}{\partial\rho}$.

3.3 Principal results for slow spot dynamics

In [39], the method of matched expansions was used to derive quasi-equilibrium solutions of (3.1) when $\epsilon \ll 1$ as follows:

Principal Result 1. *(Quasi-equilibria of (3.1) [39]). For $\epsilon \to 0$, the leading-order uniformly valid quasi-equilibrium solution to (3.1) is described by an outer solution, valid away from the spots, and inner core solutions near each of the N spots centred at $\mathbf{x} = \mathbf{x}_j$ for $j = 1, 2, \ldots, N$. These solutions are*

$$u_{\text{uinf}} \sim \epsilon^2 E + \sum_{i=1}^{N} U_{i,0}\left(\frac{|x - x_i|}{\epsilon}\right), \qquad v_{\text{unif}} \sim \sum_{i=1}^{N} S_i \log|x - x_i| - 4\pi R E + \bar{v}, \tag{3.4}$$

where $R \equiv \frac{1}{4\pi}(\log 4 - 1)$ and $\bar{v}$ is a constant. The leading-order radially symmetric inner core solution, $U_{i,0}$, is defined on the tangent plane to the sphere near the spot at $\mathbf{x} = \mathbf{x}_i$, and is found by numerical computation of the BVP (3.3). In (3.4), the spot strengths, S_i for $i = 1, 2, \ldots, N$, satisfy the nonlinear algebraic system

$$\mathcal{N}(\mathbf{S}) \equiv [\mathbf{I} - \nu(\mathbf{I} - \mathcal{E}_\mathbf{0})\mathcal{G}]\,\mathbf{S} + \nu(\mathbf{I} - \mathcal{E}_\mathbf{0})\chi(\mathbf{S}) - \frac{2E}{N}\mathbf{e} = \mathbf{0}. \tag{3.5}$$

Here $\mathbf{I}$ is the $N \times N$ identity matrix, $(\mathcal{E}_0)_{ij} = \frac{1}{N}$, $(\mathbf{S})_i = S_i$, $(\chi(\mathbf{S}))_i = \chi(S_i)$, $(\mathbf{e})_i = 1$,

$\nu = -1/\log\epsilon$, *and the Green's matrix* $\mathcal{G}$ *is*

$$(\mathcal{G})_{ij} = \begin{cases} \log|\mathbf{x}_i - \mathbf{x}_j|, & i \neq j; \\ \\ 0\,, & i = j. \end{cases} \tag{3.6}$$

The function $\chi = \chi(S_j; f)$ *is computed from the numerical solution of BVP (3.3) and the limiting process [48]*

$$\lim_{\rho\to\infty} (V_{j0} - S_j \log\rho) = \chi. \tag{3.7}$$

In terms of the spot strengths, the constant $\bar{v}$ *in (3.4) is defined as:*

$$\bar{v} = \frac{2E}{\nu N} + 4\pi RE + \frac{1}{N}\left[\mathbf{e}^{\mathbf{T}}\chi(\mathbf{S}) - \mathbf{e}^{\mathbf{T}}\mathcal{G}\mathbf{S}\right]. \tag{3.8}$$

The stability of $\mathcal{O}(1)$ time scale was discussed in detail in [39]. Some types of instabilities were derived such as self-replication and competition instabilities of the spot profile, Figures 3.1 and 3.2.

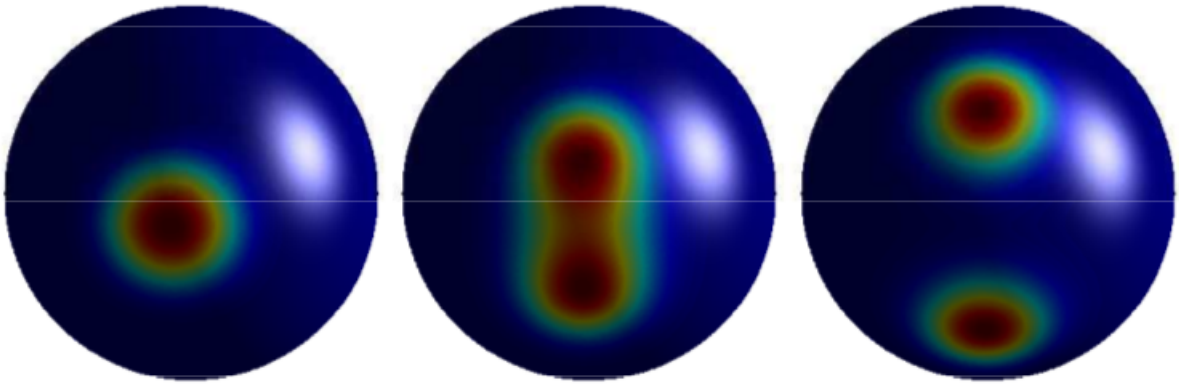

Figure 3.1: Self-replication instability for a one-spot pattern.

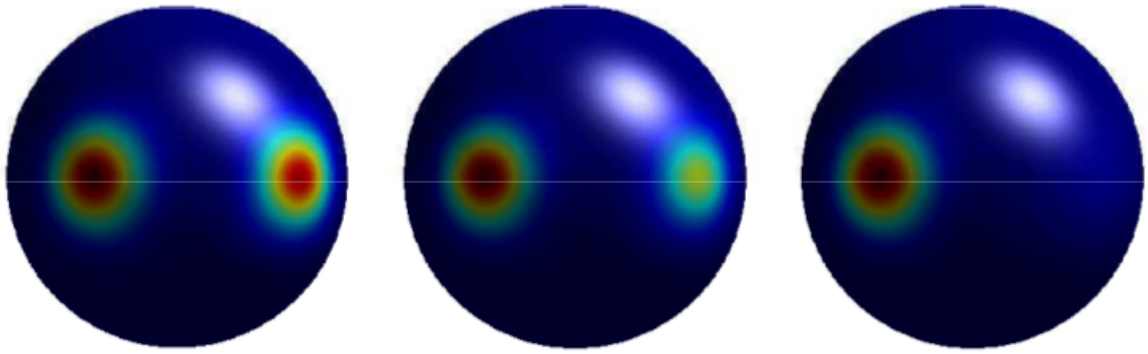

Figure 3.2: Competition instability for a two-spot pattern.

However, in case these $\mathcal{O}(1)$ time-scale instabilities are absent, the following result shows that the spots slowly spread a spot pattern on a long time scale of order $\mathcal{O}(\epsilon^{-2})$, and their motion is characterized by a system of DAEs.

Principal Result 2. *(Slow dynamics of of (3.1) [48])) Let* $\epsilon \to 0$. *Provided that there are no* $\mathcal{O}(1)$ *time-scale instabilities of the quasi-equilibrium spot pattern, the time-dependent spot locations,* $\mathbf{x}_j$ *for* $j = 1, 2, \ldots, N$, *on the surface of the sphere vary on the slow time scale* $\sigma = \epsilon^2 t$, *and satisfy the dynamics*

$$\frac{d\mathbf{x}_j}{d\sigma} = \frac{2}{\mathcal{A}_j(S_j)}(I - \mathcal{L}_j)\sum_{\substack{i=1\\ i\neq j}}^{N} \frac{\mathbf{S}_i \mathbf{x}_i}{|\mathbf{x}_i - \mathbf{x}_j|^2}, \qquad \mathcal{L}_j \equiv \mathbf{x}_j \mathbf{x}_j^T, \qquad j = 1, 2, \ldots, N, \tag{3.9a}$$

coupled to the constraints for $S_1, S_2 \ldots, S_N$ *in terms of* $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ *given by the roots of a nonlinear algebraic system involving the Green's interaction matrix* $\mathcal{G}$

$$\mathcal{N}(\mathbf{S}, \mathbf{x}) \equiv [\mathbf{I} - \nu(\mathbf{I} - \mathcal{E}_\mathbf{0})\mathcal{G})]\,\mathbf{S} + \nu(\mathbf{I} - \mathcal{E}_\mathbf{0})\chi(\mathbf{S}) - \frac{2E}{N}\mathbf{e} = \mathbf{0}. \tag{3.9b}$$

Here I *is* $N \times N$ *is identity matrix,* $(\mathcal{E}_0)_{ij} = \frac{1}{N}$, $(\mathbf{S})_i = S_i$, $(\chi(\mathbf{S}))_i = \chi(S_i)$, $(\mathbf{e})_i = 1$, $\nu = -1/\log\epsilon \ll 1$, *and* $\mathcal{G}$ *is defined in (3.6)* .

Result (2) provides a DAE system (3.9) that gathers the spot locations with spot strengths. This DAE system has index 1. It was shown that DAE system (3.9) is invariant under a rotation of spots on the unit sphere [48].

An asymptotic analysis of DAE (3.9) was performed in [48], yielding

$$\mathcal{S} \sim \frac{2E}{N}\left[\mathbf{e} + \nu(I - \mathcal{E}_0)\mathcal{G}\right] + \mathcal{O}(\nu^2)\Big]\,. \tag{3.10}$$

The functions $\chi(S_j)$ and $\mathcal{A}_j$ in System (3.9) depend only on the S_j and the Brusselator parameter f. The numerically computed function $\chi_j = \chi(S_j; f)$ is plotted in Figure 3.3.

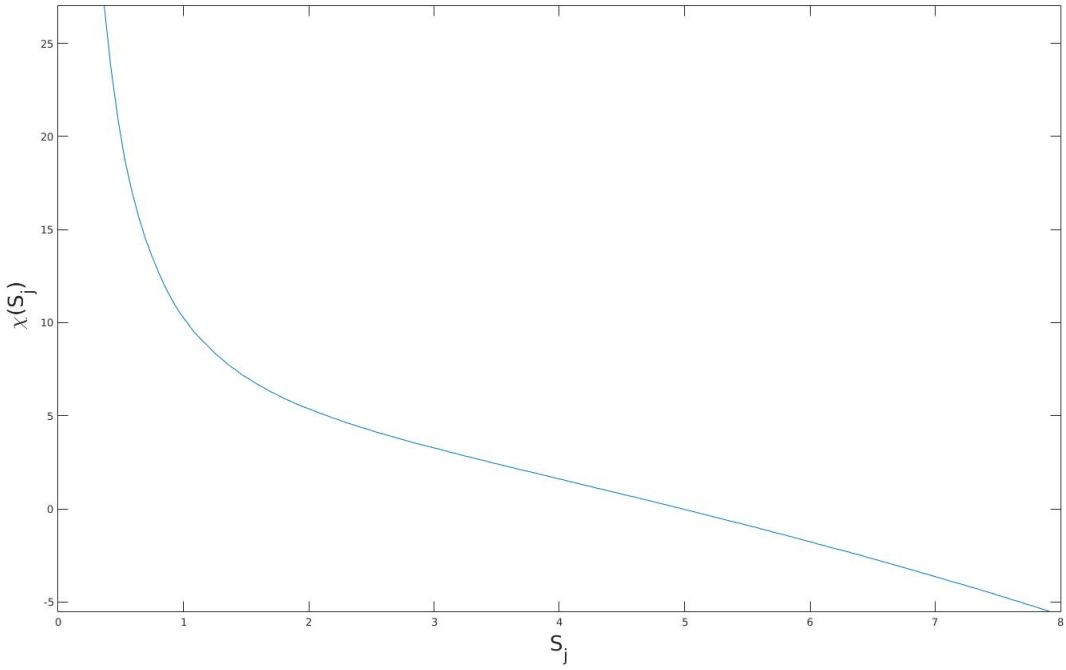

Figure 3.3: The computed function $\chi(S_j)$ versus S_j.

The function $\mathcal{A}_j = \mathcal{A}(S_j; f) < 0$ is defined by [48]

$$\mathcal{A} \equiv \int_0^\infty U'_{j0}(\rho) P_1(\rho) \rho d\rho, \tag{3.11}$$

where $U'_{j0}(\rho)$ is the solution of (3.3), and $P_1(\rho)$ satisfies

$$\Delta_\rho \begin{pmatrix} P_1(\rho) \\ \\ P_2(\rho) \end{pmatrix} - \frac{1}{\rho^2} \begin{pmatrix} P_1(\rho) \\ \\ P_2(\rho) \end{pmatrix} + \mathcal{M}_j^T \begin{pmatrix} P_1(\rho) \\ \\ P_2(\rho) \end{pmatrix} = \mathbf{0}, \tag{3.12}$$

Here, $\mathcal{M}_j$ is defined as

$$\mathcal{M}_j \equiv \begin{bmatrix} -1 + 2fU_{j0}V_{j0} & fU_{j0}^2 \\ 1 - 2U_{j0}V_{j0} & -U_{j0}^2 \end{bmatrix}. \tag{3.13}$$

The numerically computed function $\mathcal{A} = \mathcal{A}(S_j; f)$ is plotted in Figure 3.4.

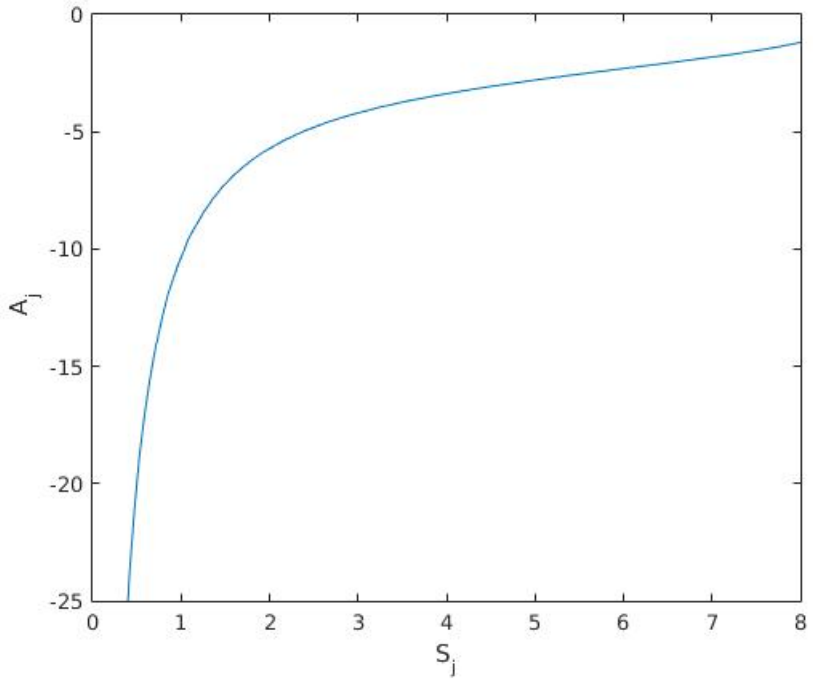

Figure 3.4: The computed function $\mathcal{A}_j$ versus S_j.

In [48], it was shown that the $\mathcal{O}(1)$ time-scale spot replication event can occur if the spot strength exceeds a threshold, $S_j > \Sigma_2(f)$. Here are threshold numerical values for a few values of f [48].

$$\Sigma_2(0.4) \approx 8.21, \quad \Sigma_2(0.5) \approx 5.96, \quad \Sigma_2(0.6) \approx 4.41, \quad \text{and} \quad \Sigma_2(0.7) \approx 3.23. \tag{3.14}$$

3.4 Numerical Solutions of the DAE system

In this work, we solve the DAE system (3.9) numerically as follows.

i. N initial spots locations are generated.

Let h_ϕ and h_θ be uniformly distributed random variables in $(0, 1)$. The spherical coordinates can be defined as

$$\theta = 2\pi h_\theta\,, \qquad \phi = \cos^{-1}(2h_\phi - 1), \tag{3.15}$$

then transform spherical coordinates to Cartesian.

ii. Using the spot locations from (i), N guesses for spot strengths S_j that satisfy the equation (3.10) are obtained.

iii. Matlab's `fsolve` function is used to solve the algebraic system (3.9b) for the spot strengths S_j given the initial S_j obtained from (ii).

iv. Matlab's `ode15s` function is used to solve the DAE system (3.9) with the initial conditions obtained from steps (ii) and (iii).

We remark that if the convergence fails for any used Matlab functions, we simply generate a new random initial set for spot locations in step (i).

Numerical computations for $N = 6$ spots

As an example, we illustrate the numerical solution of the case of $N = 6$ spots. For this computation, we took $f = 0.5$, $\epsilon = 0.02$, and $E = 12$. Figure 3.5 shows the evolution of 6 spots. The numerical results for the spot strengths are $S_j = 4, \quad j = 1, 2, \ldots, 6$.

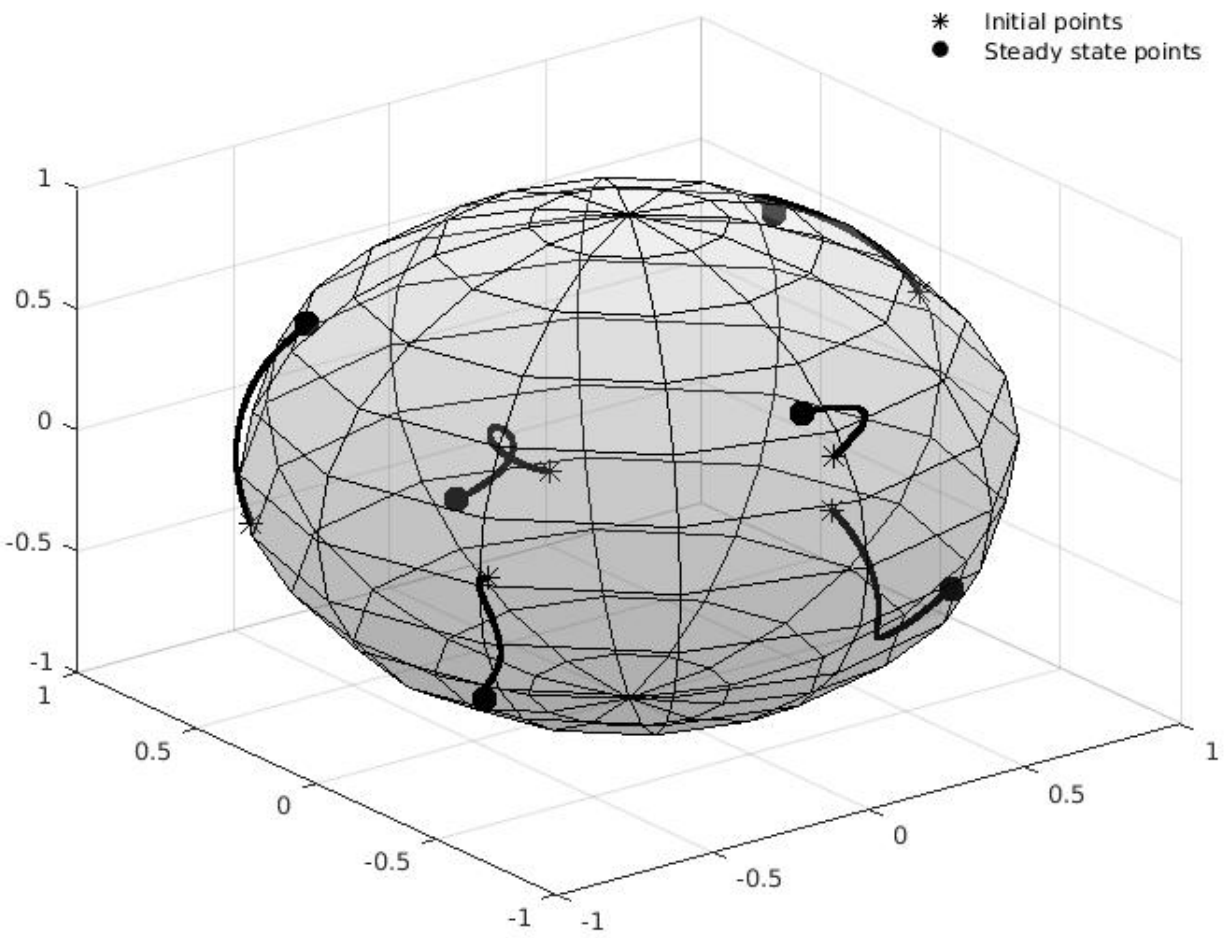

Figure 3.5: The evolution of 6 spots pattern for the Brusselator model (3.1).

Chapter 4

An Overview of Particle swarm optimization

4.1 Introduction

Numerical optimization is a field of applied mathematics and numerical analysis. Simply put, it is the approach of finding the best (maximum or minimum) value(s) of an objective function according to a set of criteria. The objective function can be time, profit, distance, potential energy, etc. Optimization problems arise in many areas of science, engineering, and economics.

This chapter reviews some basic concepts related to the optimization, presents some types of optimization problems, and discusses the particle swarm optimization method [25].

The continuous optimization problem can be summarized as follows

$$\begin{aligned} &\text{Given } f : \mathbb{R}^n \to \mathbb{R}, \\ &\text{find } \mathbf{x}^* \in \mathbb{R}^n \text{ such that, } f(\mathbf{x}^*) \leq f(\mathbf{x}),\ \forall \mathbf{x} \in \mathbb{R}^n. \end{aligned} \tag{4.1}$$

It is obvious that if a point $\mathbf{x}$ in (4.1) represents the minimum value of an objective function $f(\mathbf{x})$, then it represents the maximum value of the objective function $-f(\mathbf{x})$.

Example 4.1.1. *The Sphere function is given as [12]*

$$f(\mathbf{x}) = \sum_{i=1}^{n} x_i^2 \tag{4.2}$$

where n is the number of dimensions. This function has a global minimum 0 at $x_i = 0$ for

$i = 1, 2, \ldots, n$; *see Figure 4.1.*

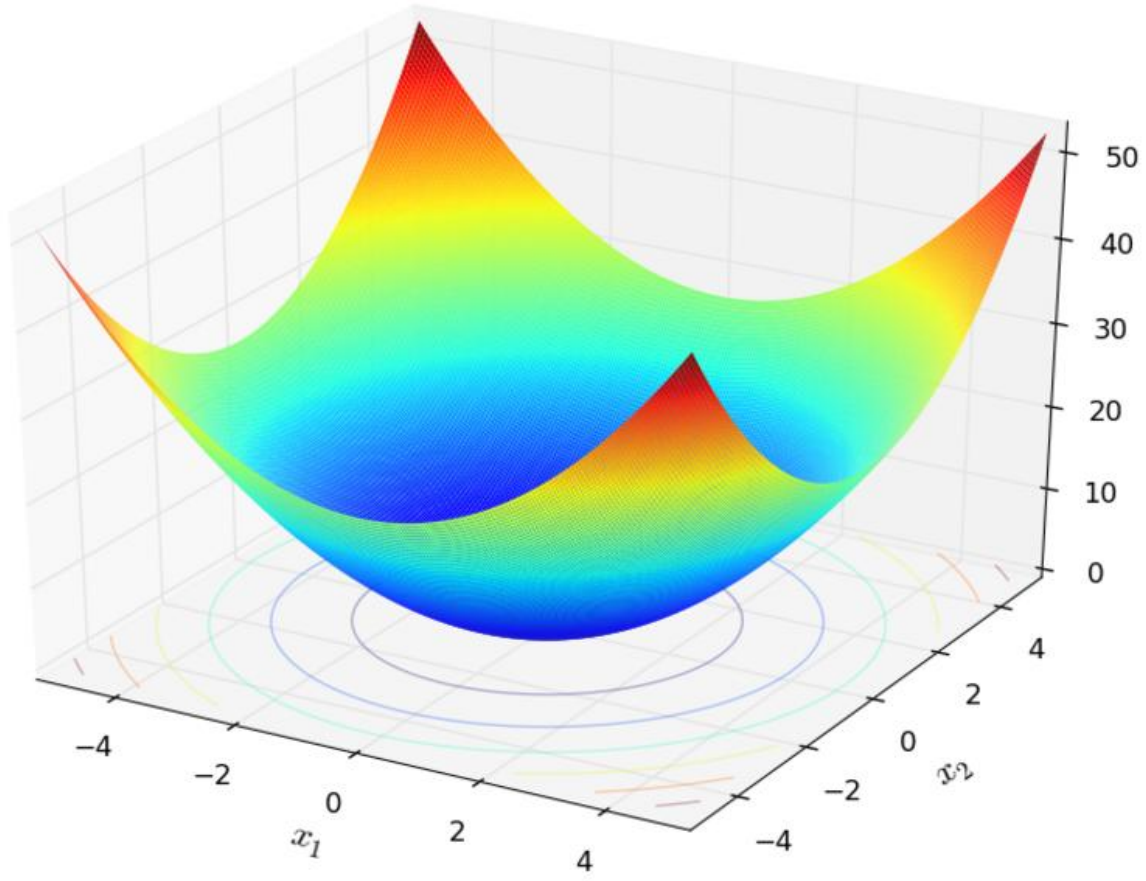

Figure 4.1: Two-dimensional Sphere function

4.2 Classifications of optimization problems

The field of optimization is very popular, and it has many theories, methods, and software. Classifying optimization methods helps to understand optimization problems and choosing appropriate algorithms for their solution. Here, we present some types of optimization problems [16, 18]

- Continuous versus discrete optimization

  If the optimization problem has a continuous domain, like a set of real numbers, then it is a continuous optimization problem. On the other hand, if the domain of the

optimization problem is a set of discrete values such as the integers, then it is a discrete optimization problem.

- Constrained versus unconstrained optimization

  Constrained optimization is the problem of optimizing an objective function subjected to a set of constraints. It can be presented as follows:

$$\begin{aligned} &\text{find } \mathbf{x} \text{ for which } f(\mathbf{x}) \text{ is minimized} \\ &\text{subject to} \\ &g_i(\mathbf{x}) = c_i \quad \text{for } i = 1, 2, \ldots, l, \ \text{(equality constraints)} \\ &h_j(\mathbf{x}) = c_i \quad \text{for } j = 1, 2, \ldots, m \, \text{(inequality constraints)}. \end{aligned} \tag{4.3}$$

  If the optimization problem does not include any constraints then it is unconstrained optimization problem, such as problem (4.1).

- Linear versus non-linear optimization

  A linear optimization problem is an optimization problem that has a linear objective function and all its constraints are linear functions. If the objective function or any of the constraints is non-linear, then it is a non-linear optimization problem.

- Global versus local optimization

  The aim of a global optimization problem is to find the globally optimal solution over its domain, whereas a local optimization problem finds an optimal solution that is dependent on the value at which the algorithm is started.

- Single versus multi-objective optimization

  A single optimization problem is an optimisation problem with a single objective function. If the optimization problem involves more than one objective function problem then it is a multi-objective optimization problem.

4.3 Particle swarm optimization

Particle Swarm Optimization (PSO) is a technique created by James Kennedy and Russell C. Eberhart in 1995 [25]. The PSO method has been widely used in many applications, and it has achieved tremendous success. In 2008, it was shown that more than eleven hundred papers stored in the IEEE Xplore database used PSO methods [34].

The concept of this technique is based on the social habits of bird flocking or fish schooling. When a set of birds looks for food from one area to another, usually there is a bird that can identify the food source very well, and then all the group follows this bird, which is nearest to the food. In terms of PSO, each single solution is called a particle in the search space. All particles have numerical fitness values that are computed by the objective function to be optimized.

A particle consists of

- a position in the search space,
- the fitness value at this position,
- a velocity,
- personal best value (pBest), which is the best fitness the particle ever obtained,
- global best value (gBest), which is the best fitness achieved so far among all particles in the space.

The group of particles is called a swarm. The search for the optimal solution is performed systematically. The PSO algorithm performs a number of iterations, each iteration consists of three steps:

1. compute the fitness values of each particle,
2. generate the personal and global optimal fitnesses and positions,
3. update the velocity and position of each particle.

After finding the pBest and gBest values, one can update the particle's velocity and position using the following iteration, [25]:

$$\mathbf{v}^i(k+1) = \mathbf{v}^i(k) + 2\mathbf{r}_1^i(k)\left[\mathbf{x}_p^i(k) - \mathbf{x}^i(k)\right] + 2\mathbf{r}_2^i(k)\left[\mathbf{x}_g(k) - \mathbf{x}^i(k)\right], \qquad k = 0, 1, \ldots, K-1, \tag{4.4}$$

$$\mathbf{x}^i(k+1) = \mathbf{x}^i(k) + \mathbf{v}^i(k+1), \qquad k = 0, 1, \ldots, K-1, \tag{4.5}$$

where $\mathbf{x}^i(k)$ is the position of particle i at iteration k, K is the total number of iterations, and $\mathbf{v}^i(k)$ is the velocity of particle i at iteration k. Here $\mathbf{x}_p$ is the personal best position, and $\mathbf{x}_g$ is the global best position. The vectors $\mathbf{r}_1$ and $\mathbf{r}_2$ contain random numbers uniformly distributed between 0 and 1.

The flowchart (4.2) shows the PSO algorithm.

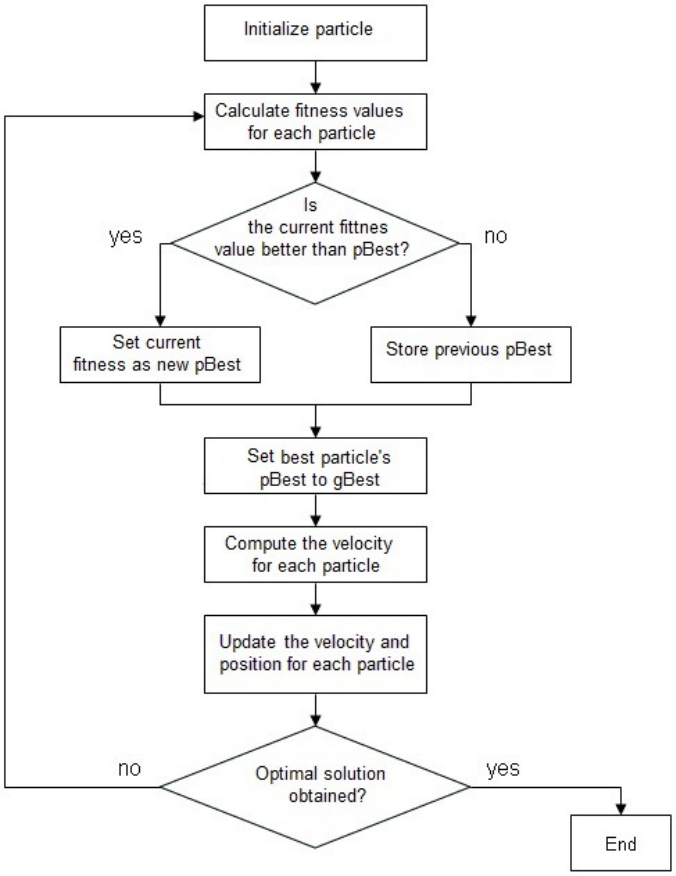

Figure 4.2: PSO algorithm

4.3.1 Advantages of the PSO algorithm

PSO is considered to be one of the most popular optimization methods, and it has many advantages as follows [11, 13, 19, 29]:

- PSO is a simple technique. The PSO computations can be implemented easily compared with other optimization methods.
- PSO is a derivative-free method.
- There are not many parameters in PSO, and their effects on the solution is limited.
- The impact of a set of initial points is small compared to other methods; this gives an advantage to convergence of the algorithm.
- PSO is less sensitive to the nature of the objective function.
- Overlapping and mutation calculations are avoided in the PSO algorithm, making the speed of the search fast.

On the other hand, PSO has some limitations. The randomness aspects in the PSO algorithm and the inefficient performance in some applications [29] can be considered as limitations.

4.3.2 Some variants of PSO

In order to get satisfactory performance, the classical PSO algorithm has been subjected to many updates over the last decade. In those variants, the way of calculating the velocity and position was improved, and the way of selecting the PSO parameters was designed to avoid the divergence of PSO solutions.

Here we present some PSO variants.

Discrete particle swarm optimization

Most PSO algorithms work on a continuous space, but the search space in discrete particle swarm optimization (DPSO) is discrete, so the particle positions are assigned to discrete

values. The most common form of DPSO is binary optimization, which was introduced by Kennedy and Eberhart in 1997 [26] to work in binary spaces. In binary optimization, the velocity is updated as in classical PSO (using (4.4)), but the position is updated by the following rule,

$$\mathbf{x}^i(k+1) = \begin{cases} 1, & \text{if } r < S(\mathbf{v}_i(k+1)) \\ 0, & \text{otherwise,} \end{cases} \tag{4.6}$$

where $S(x)$ is the sigmoid function given by

$$S(x) = \frac{1}{1+e^{-x}} \tag{4.7}$$

and the random numbers in $\mathbf{r}$ are distributed uniformly on $[0, 1]$.

Global best particle swarm optimization

Global best PSO (GBPSO) was introduced in [25] with the following change in the velocity update:

$$\mathbf{v}^i(k+1) = w\mathbf{v}^i(k) + c_1\mathbf{r}_1^i(k)\left[\mathbf{x}_p^i(k) - \mathbf{x}^i(k)\right] + c_2\mathbf{r}_2^i(k)\left[\mathbf{x}_g(k) - \mathbf{x}^i(k)\right] \tag{4.8}$$

$$k = 0, 1, \ldots, K-1, \tag{4.9}$$

where w is a parameter called inertia weight and c_1 and c_2 are called acceleration coefficients.

The inertia weight w plays a significant role in the convergence of PSO algorithm; the classical PSO has no inertia weight. The concept of inertia weight was presented for the first time in 1998 by Shi and Eberhart [43].

Decreasing weight particle swarm optimization

Decreasing weight particle swarm optimization (DWPSO) provides a new formula for the inertia weight, which is decreased linearly over time [15]. It is defined as follows:

$$w(k) = w_s - (w_s - w_e)\frac{k}{K}, \tag{4.10}$$

where w_s is the inertia weight set for the first iteration, and w_e is the inertia weight set for the last iteration K.

In DWPSO, the velocities are calculated using the following iteration

$$\begin{aligned} \mathbf{v}^i(k+1) =& w(k)\mathbf{v}^i(k) + c_1\mathbf{r}_1^i(k)\left[\mathbf{x}_p^i(k) - \mathbf{x}^i(k)\right] + c_2\mathbf{r}_2^i(k)\left[\mathbf{x}_g^i(k) - \mathbf{x}^i(k)\right] && (4.11)\\ & k = 0, 1, \ldots, K-1, && (4.12) \end{aligned}$$

One can note from (4.9) that the inertia weight w at every iteration k depends on the total number of iterations K, so changing the number iterations K generally leads to different results at every iteration.

Time-varying acceleration coefficients PSO

Time-varying acceleration coefficients PSO (TVACPSO) provides some modifications not only on the inertia weight w but also on the acceleration coefficients c_1 and c_2 [15, 37]. The inertia weight w is changed as in DWPSO using (4.9) at every iteration k. TVACPSO uses the following iteration to obtain the velocities

$$\begin{aligned} \mathbf{v}^i(k+1) =& w(k)\mathbf{v}^i(k) + c_1(k)\mathbf{r}_1^i(k)\left[\mathbf{x}_p^i(k) - \mathbf{x}^i(k)\right] + c_2(k)\mathbf{r}_2^i(k)\left[\mathbf{x}_g(k) - \mathbf{x}^i(k)\right] && (4.13)\\ & k = 0, 1, \ldots, K-1, && (4.14) \end{aligned}$$

$c_1(k)$ is the personal best weight at iteration k and $c_2(k)$ is the global best weight at iteration k; they are obtained using the following equations:

$$c_1(k) = c_{1s} - (c_{1s} - c_{1e})\frac{k}{K}, \tag{4.15a}$$

$$c_2(k) = c_{2s} - (c_{2s} - c_{2e})\frac{k}{K}, \tag{4.15b}$$

where c_{1s} is the personal best weight assigned for the first iteration, c_{1e} is the personal best weight assigned for the last iteration K, c_{2s} is the global best weight designated for the first iteration, and c_{2e} is the global best weight designated for the last iteration K.

In TVACPSO, the total number of iterations K has a significant role. It affects the values of w, c_1, and c_2 at every iteration.

Other PSO variants

Other formulas for the inertia weight w were developed by many researchers. Table 4.1 presents some different formulas for inertia weight.

Name of inertia weight	Formula	Reference
Constant inertia weight	$w = \text{constant}$	[43]
Random inertia weight	$w = 0.5 + \frac{Rand()}{2}$	[14]
Global-local best inertia weight	$w(k) = 1.1 - \frac{gBest(k)}{pBest(k)}$	[3]
Simulated annealing inertia weight	$w(k) = w_{\min} + (w_{\max} - w_{\min})\lambda^{(k-1)},\ \lambda = 0.95$	[2]
Linear decreasing inertia weight	$w(k) = w_{\max} + (w_{\min} - w_{\max})\log_{10}(a + \frac{10k}{K})$, a is a constant	[20]

Table 4.1: Different types of inertia weight

A large amount of literature about the PSO theory, its variants, and its applications is available, e.g., [7, 22, 24, 41, 43, 46, 47, 50, 53]

Chapter 5

Optimization Results

5.1 Introduction

The key idea in this chapter is to use the connection between the numerical solutions of DAE (3.9) and elliptic Fekete points. A set of points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ on the unit sphere that globally minimizes the logarithmic energy

$$\mathcal{H}(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n) = -\sum_{i=1}^{N} \sum_{j=i+1}^{N} \log |\mathbf{x}_i - \mathbf{x}_j|, \tag{5.1}$$

is called an elliptic Fekete points set [17]. The classical Fekete problem was studied widely in various sources, e.g., [5, 17, 44, 45, 49]. It was shown in [48] that elliptic Fekete point sets are indeed equilibrium spot configurations of the DAE (3.9). Therefore, equilibrium spot configurations can be obtained by solving the optimization problem (5.1). In this chapter, the GBPSO method is used to obtain the minimal values of logarithmic energy (5.1) for $N = 2, 3, \ldots, 19$ spots. The GBPSO method is implemented in the `pyswarm` python package [1]. The PSO parameters are chosen to be $w = 0.85$ and $c_1 = c_2 = 0.5$. The localized and strength patterns for $N = 2, 3, \ldots, 12$ Fekete points are presented. The relationship between the equilibria of the DAE and the Fekete points is discussed.

5.2 Minimal values of logarithmic energy

On a unit sphere, any point can be represented in terms of spherical coordinates (θ, ϕ); see Figure 5.1. To avoid the effect of the rotational symmetries of the sphere, it is convenient to fix the first spot $\mathbf{x}_1$ at the north pole, i.e., $(\theta_1, \phi_1) = (0, 0)$, and let $\theta_2 = 0$ for the second spot

$\mathbf{x}_2$. Then, for N spots on the unit sphere, we have a global optimization problem of $2N-3$ parameters in the range $0 \leq \phi_j \leq \pi$ for $j = 2, 3, \ldots, N$, and $0 \leq \theta_j \leq 2\pi$ for $j = 3, 4, \ldots, N$.

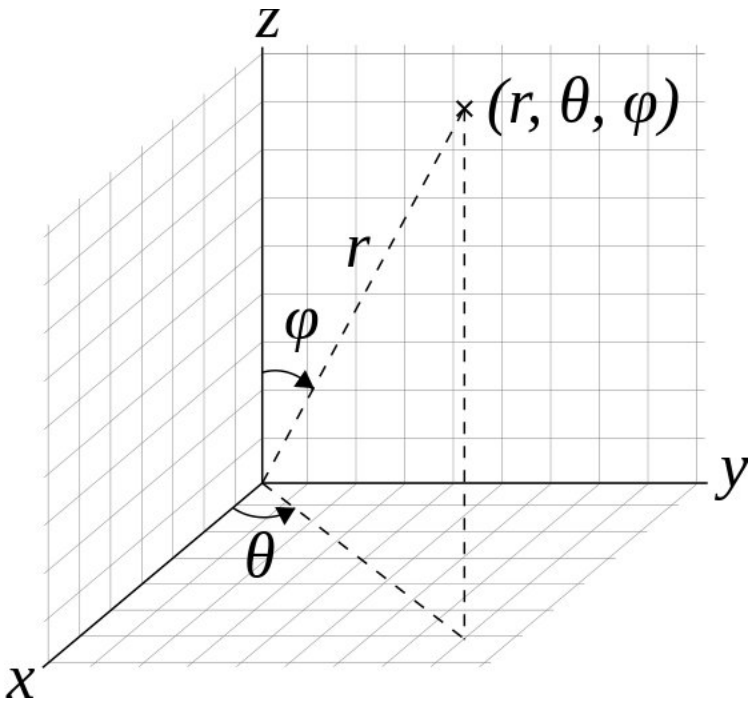

Figure 5.1: The spherical coordinate of a point

We present the optimal values of the function (5.1) for N spots arranged on the unit sphere, as outlined above. Table 5.1.

N	$\mathcal{H}$	N	$\mathcal{H}$
2	-0.693147	11	-18.420476
3	-1.647918	12	-21.606145
4	-2.942488	13	-24.866718
5	-4.420507	14	-28.407811
6	-6.238325	15	-32.147872
7	-8.182477	16	-36.105923
8	-10.428018	17	-40.273062
9	-12.887753	18	-44.650281
10	-15.563123	19	-49.199821

Table 5.1: Minimal values of logarithmic energy

We remark that our results match with results of [5] for the logarithmic energy.

5.3 Elliptic Fekete point sets

In this section, we present the Fekete points for $N = 2, 3, \ldots, 12$. The localized and strength patterns are presented. Matlab's `fsolve` function is used to obtain the spot strengths S_j from the algebraic system (3.9b). Here, we set $f = 0.05$, $\epsilon = 0.02$, and $E = 2N$.

Two spots

The spherical coordinates of the elliptic Fekete points of $N = 2$ spots are shown in Table 5.2. The Fekete points are located at the poles, as shown in Figure 5.2, with equal spot strengths, $(S = 4)$.

	Spherical coordinates	
θ	0	0
ϕ	0	3.1416

Table 5.2: Spherical coordinates of 2 Fekete points

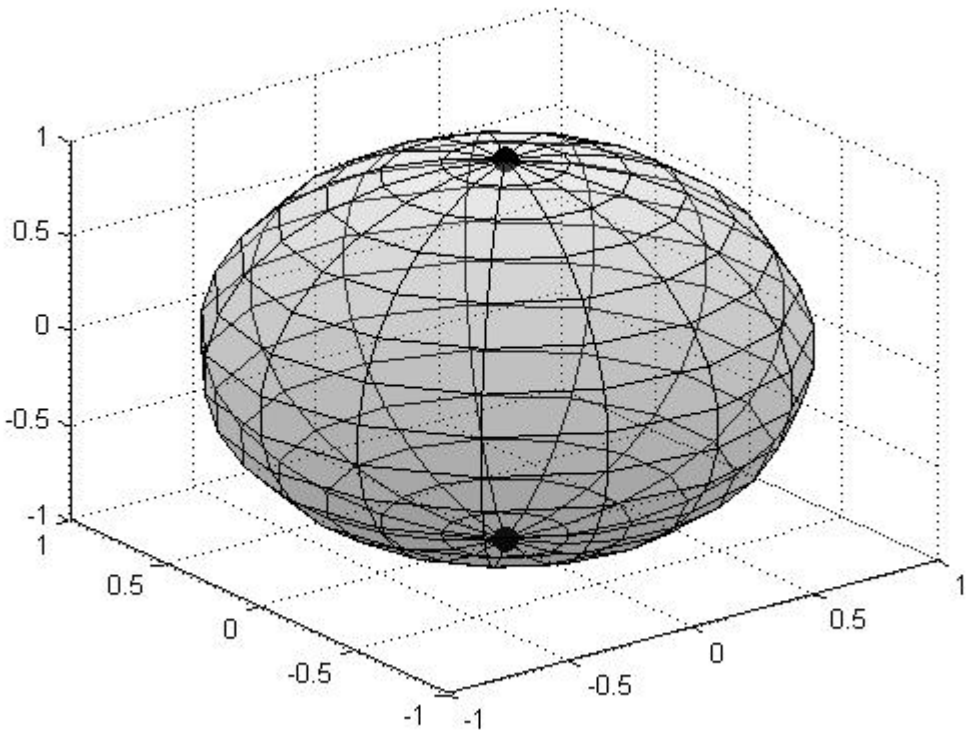

Figure 5.2: Elliptic Fekete points for $N = 2$

Three spots

The spherical coordinates of the elliptic Fekete points of $N = 3$ spots are shown in Table 5.3. The Fekete points occupy vertices of an inscribed equilateral triangle, as shown in Figure 5.3. The spot strengths are equal, $(S = 4)$.

	Spherical coordinates		
θ	0	0	3.1416
ϕ	0	2.0944	2.0944

Table 5.3: Spherical coordinates of 3 Fekete points

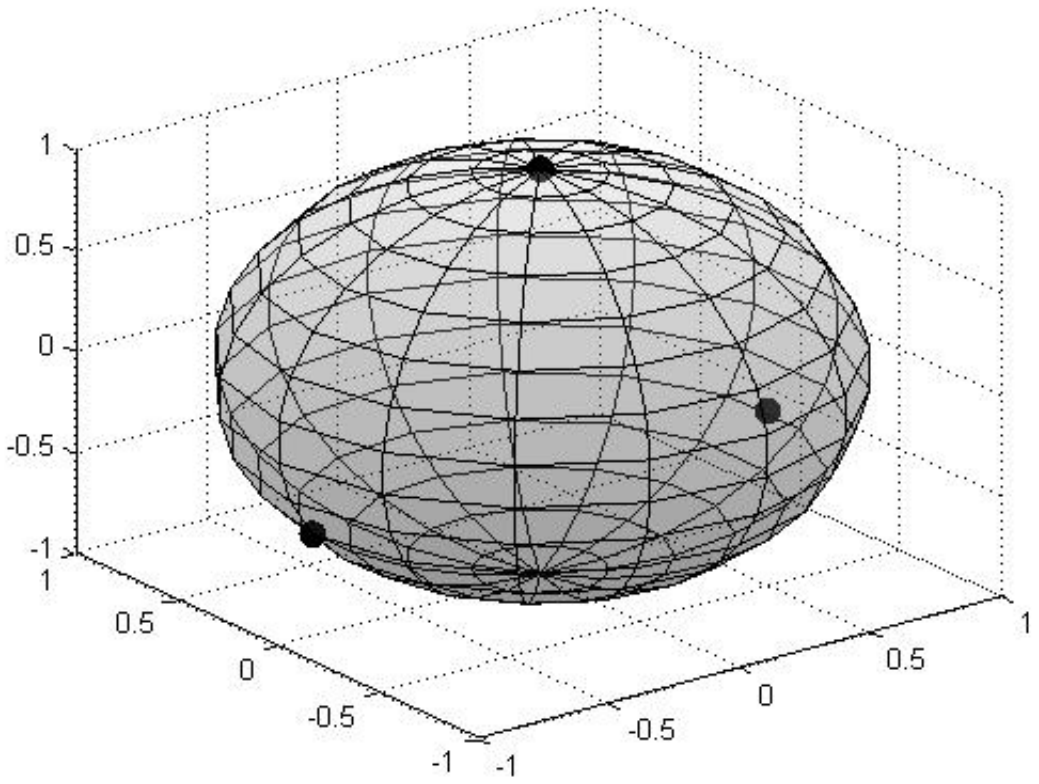

Figure 5.3: Elliptic Fekete points for $N = 3$

Four spots

The spherical coordinates of the elliptic Fekete points of $N = 4$ spots are shown in Table 5.4. The Fekete points are located on the vertices of an inscribed tetrahedron, as shown in Figure 5.4, with a common spot strength pattern ($S = 4$).

	Spherical coordinates			
θ	0	0	4.1888	2.0944
ϕ	0	1.9106	1.9106	1.9106

Table 5.4: Spherical coordinates of 4 Fekete points

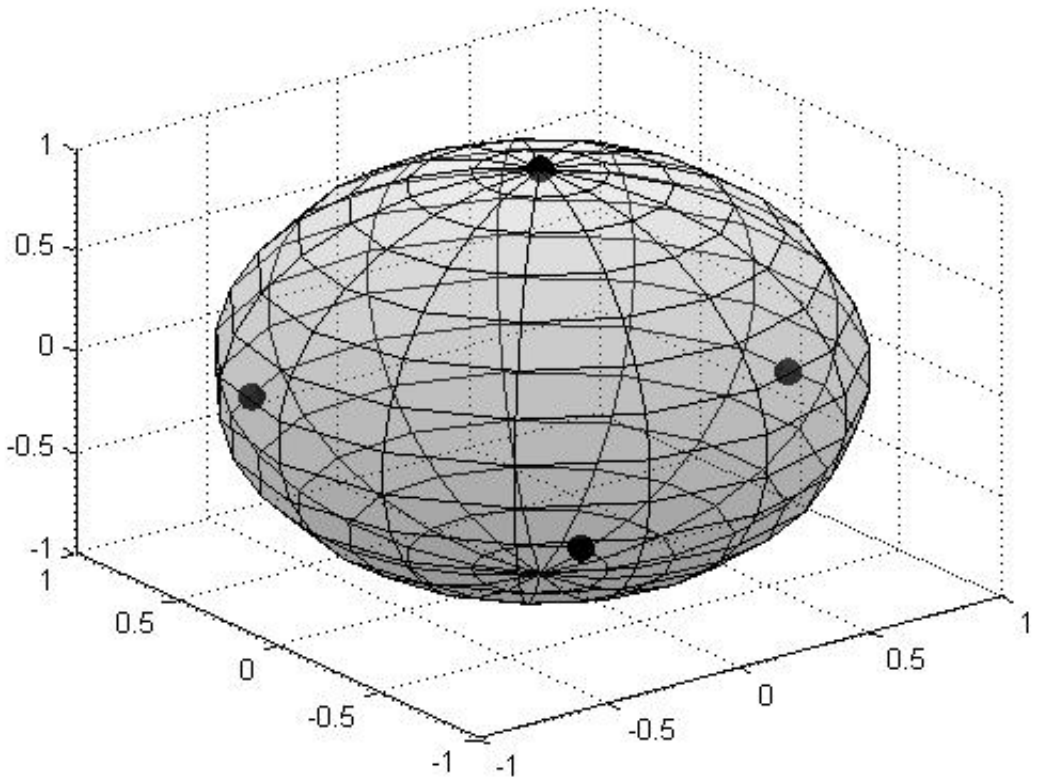

Figure 5.4: Elliptic Fekete points for $N = 4$

Five spots

The spherical coordinates of the elliptic Fekete points for $N = 5$ are shown in Table 5.5. We obtain two antipodal spots, while the other three spots are located on the mid-plane, as shown in Figure 5.5. The spot strengths for the spots in the polar are different than the points in the mid-plane, $S_1 = S_2 = 3.94$ and $S_j = 4.04$, for $j = 3, 4, 5$.

	Spherical coordinates				
θ	0	0	0	4.1889	2.0944
ϕ	0	3.1416	1.5707	1.5708	1.5709

Table 5.5: Spherical coordinates of 5 Fekete points

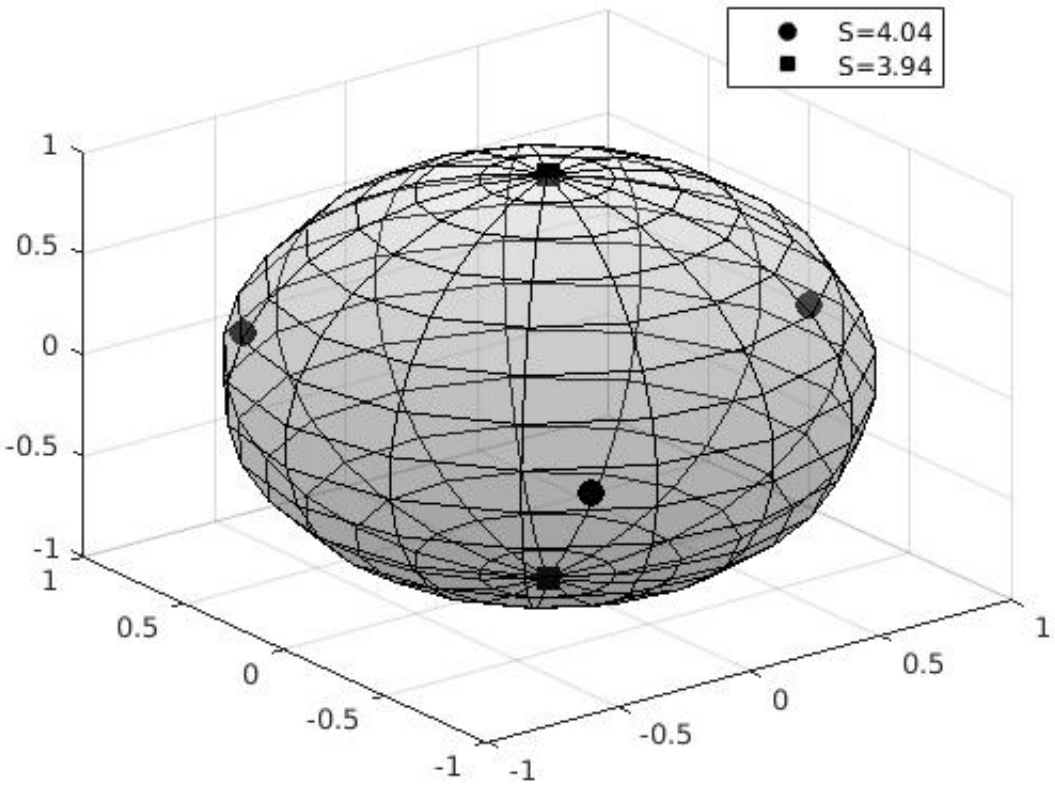

Figure 5.5: Elliptic Fekete points for $N = 5$

Six spots

The spherical coordinates of the elliptic Fekete points for $N = 6$ are shown in Table 5.6. The localized pattern consists of two antipodal spots, while the other four spots are located on the mid-plane, see Figure 5.6. The spot strengths for all spots are equal ($S = 4$).

	Spherical coordinates					
θ	0	0	5.9963	4.7124	1.5708	3.1416
ϕ	0	1.5708	3.1416	1.5708	1.5708	1.5708

Table 5.6: Spherical coordinates of 6 Fekete points

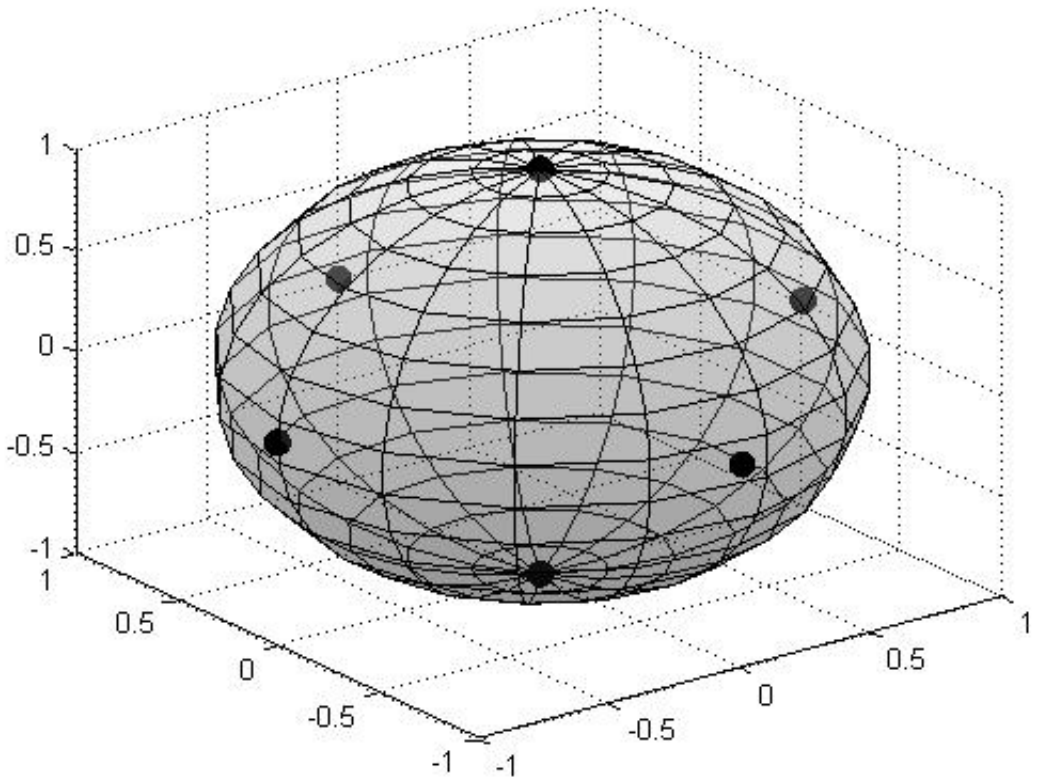

Figure 5.6: Elliptic Fekete points for $N = 6$

Seven spots

The spherical coordinates of the elliptic Fekete points for $N = 7$ are shown in Table 5.6. The localized pattern consists of two antipodal spots, with a common strength ($S = 4.15$), four spots are located on two parallel planes, and one terminal spot, with a common strength ($S = 3.94$), as shown in Figure 5.7.

	Spherical coordinates						
θ	0	0	-1.2573	1.2575	-2.5135	0	2.5139
ϕ	0	1.5675	1.5566	1.5892	1.5966	3.1402	1.5442

Table 5.7: Spherical coordinates of 7 Fekete points

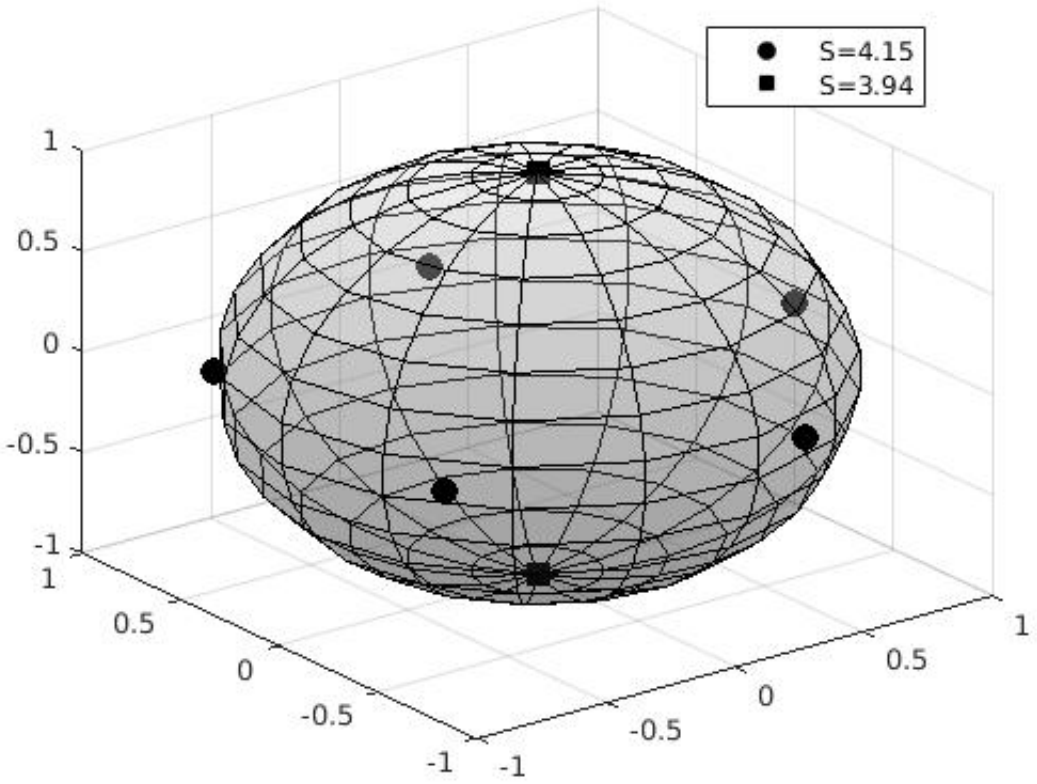

Figure 5.7: Elliptic Fekete points for $N = 7$

Eight spots

The spherical coordinates of $N = 8$ elliptic Fekete points are shown in Table 5.8, and the spots localized patterns are shown in Figure 5.8. It can be described as three parallel planes of two spots each, one northern pole and one terminal spot. The spot strengths for all spots are equal ($S = 4$).

	Spherical coordinates							
θ	0	0	4.3091	0.7070	5.0170	3.5648	1.4519	2.5083
ϕ	0	1.4068	2.4987	2.4989	1.4071	1.2463	1.2465	1.9422

Table 5.8: Spherical coordinates of 8 Fekete points

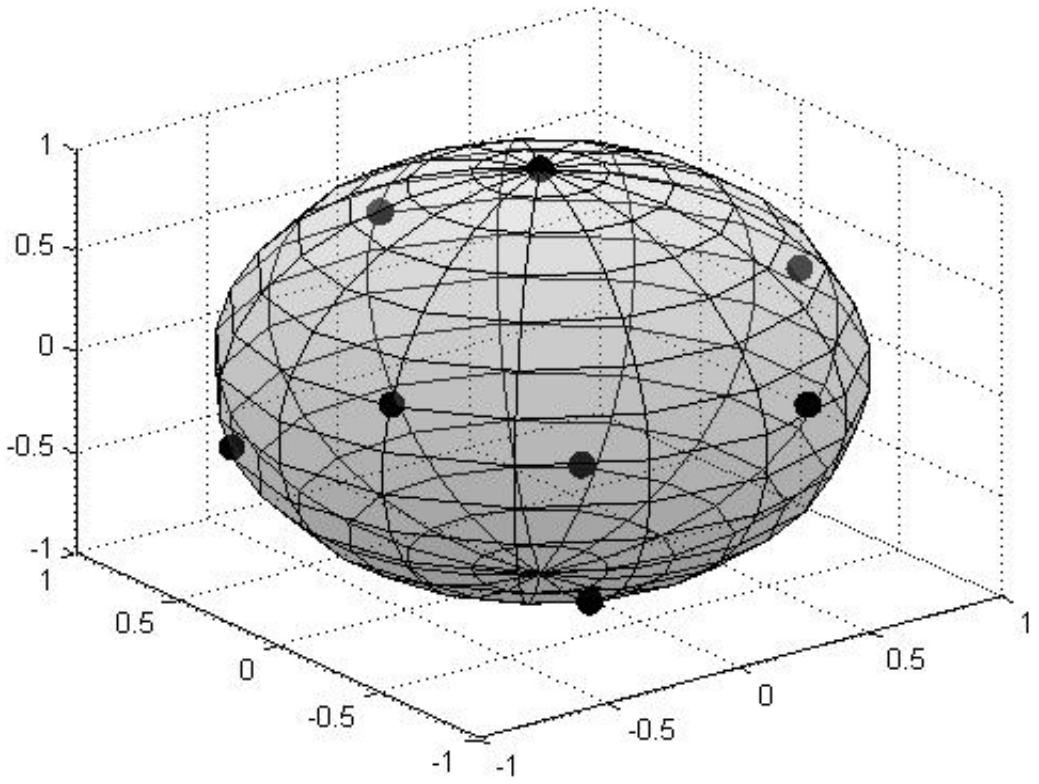

Figure 5.8: Elliptic Fekete points for $N = 8$

Nine spots

The spherical coordinates of $N = 9$ elliptic Fekete points are shown in Table 5.9. The localized spot pattern consists of a northern pole and three parallel planes, one contains four spots and the others of two spots each,The spots localized and strength patterns are shown in Figure 5.9.

	Spherical coordinates								
θ	0	0	5.0985	1.1860	5.0974	3.1411	3.8289	1.1838	2.4544
ϕ	0	1.5584	2.4143	1.2077	1.2073	2.3624	1.3274	2.4146	1.3270

Table 5.9: Spherical coordinates of 9 Fekete points

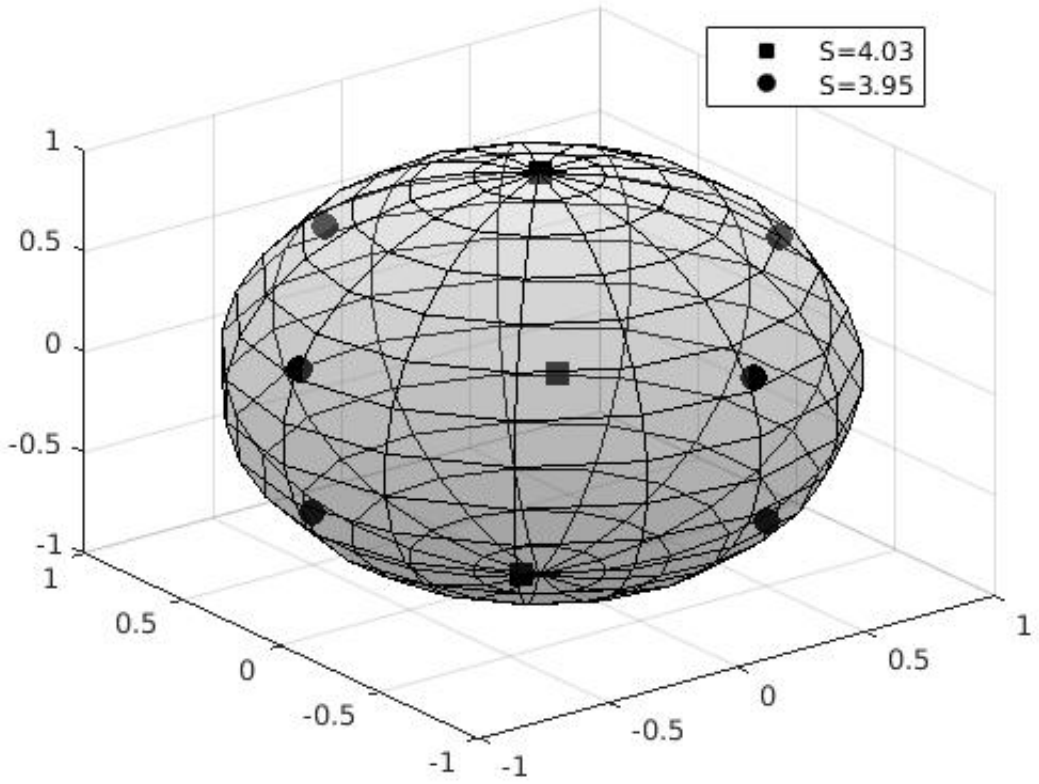

Figure 5.9: Elliptic Fekete points for $N = 9$

Ten spots

The spherical coordinates of $N = 10$ elliptic Fekete points are shown in Table 5.10. The localized spots pattern consists of two antipodal spots, with a common strength ($S = 3.94$), and two parallel planes with four spots on each, with a common strength ($S = 4.02$), as shown in Figure (5.10).

	Spherical coordinates									
θ	0	0	1.5710	0	5.4976	2.3562	4.7124	0.7851	3.1412	3.9268
ϕ	0	1.1368	1.1369	3.1416	2.0048	2.0052	1.1365	2.0047	1.1368	2.0046

Table 5.10: Spherical coordinates of 10 Fekete points

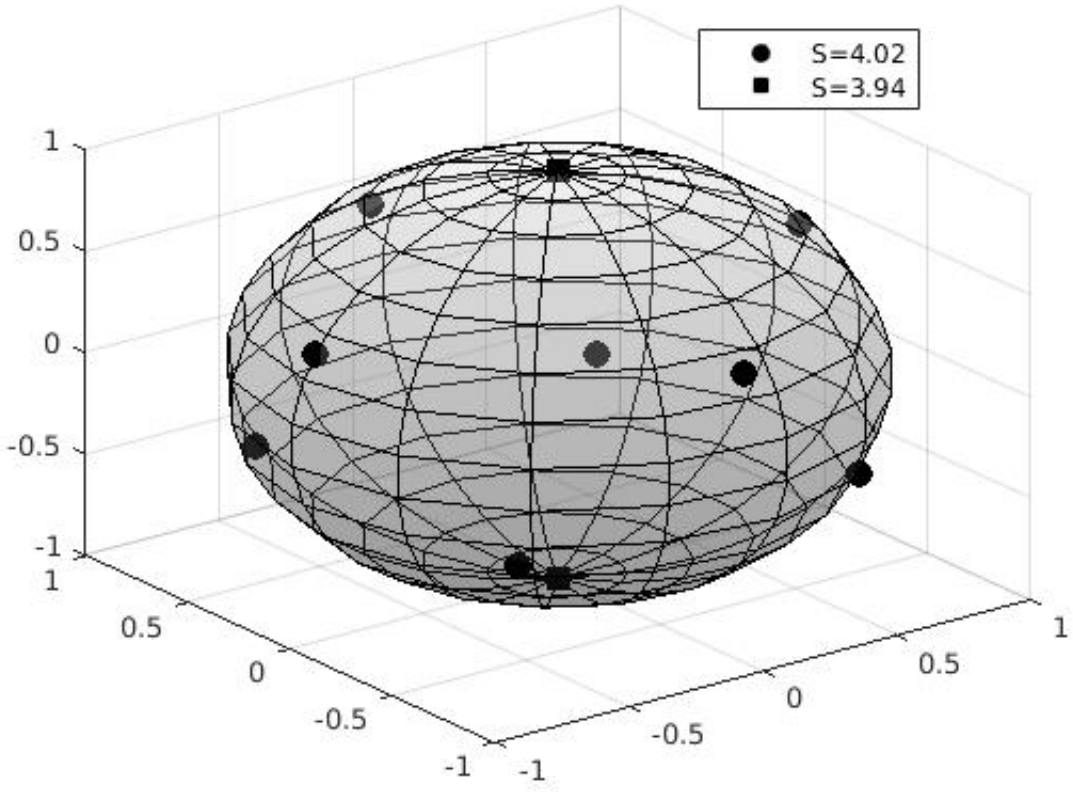

Figure 5.10: Elliptic Fekete points for $N = 10$

Eleven spots

The spherical coordinates of $N = 11$ elliptic Fekete points are shown in Table 5.11. The irregular spots localized pattern and spots strengths pattern are shown in Figure (5.11).

	Spherical coordinates										
θ	0	4.8039	1.2774	0	2.2844	4.1273	3.4625	2.7449	0	5.3759	1.0424
ϕ	0	1.0140	2.0582	2.7861	1.2578	2.0999	1.1971	2.2779	1.3939	1.8978	1.0424

Table 5.11: Spherical coordinates of 11 Fekete points

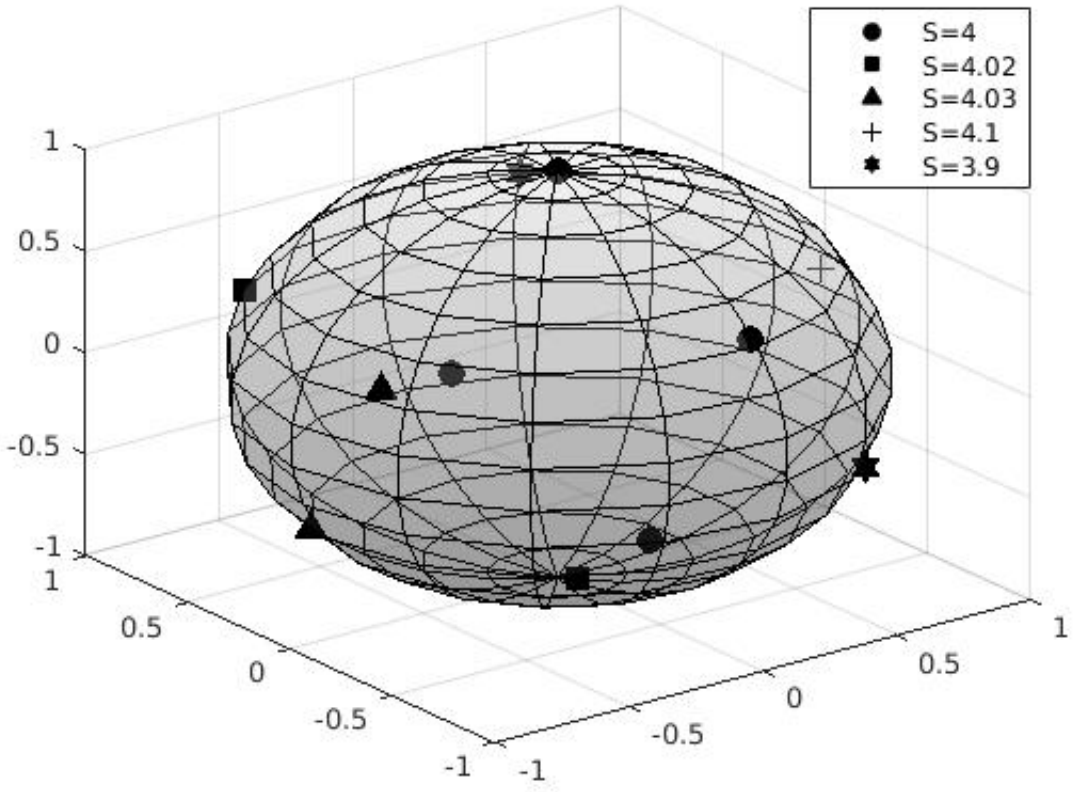

Figure 5.11: Elliptic Fekete points for $N = 11$

Twelve spots

The spherical coordinates of $N = 12$ elliptic Fekete points are shown in Table 5.12. The localized spot pattern consists of two antipodal and two parallel planes with five spots each, as shown in Figure 5.12. The spot strengths for all spots are equal ($S = 4$).

	Spherical coordinates											
θ	0	0	1.8850	4.3982	3.1416	5.0267	3.7697	2.5131	0.6283	6.2832	5.6545	1.2564
ϕ	0	1.1074	2.0344	2.0342	2.0345	1.1070	1.1071	1.1071	2.0346	3.1416	2.0345	1.1071

Table 5.12: Spherical coordinates of 12 Fekete points

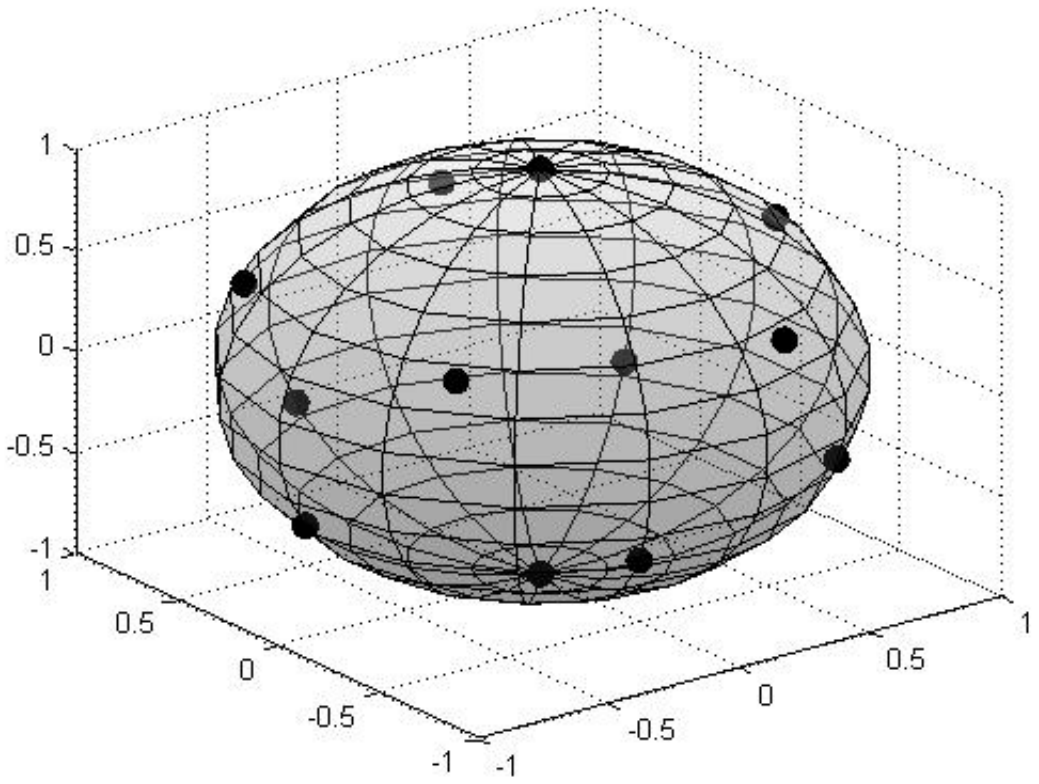

Figure 5.12: Elliptic Fekete points for $N = 12$

5.4 More about the relationship between equilibria of the DAE and the Fekete points

Our numerical solutions of the DAE system for $N = 2, 3, \ldots, 8$ spots starting from a set of initial conditions can lead to three cases:

- Steady-state Fekete points

  We obtain a set of initial conditions that converge to elliptic Fekete points; Figure 5.13. The values of the logarithmic energy during the evolution of these spots are shown in Figure 5.14.

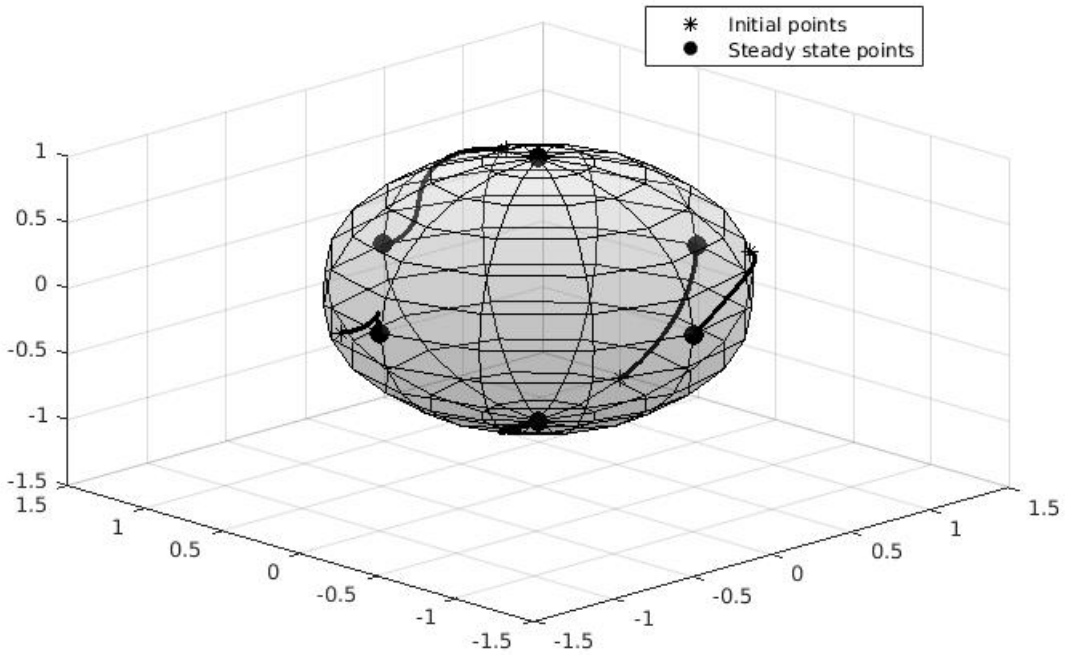

Figure 5.13: Fekete Steady-State points

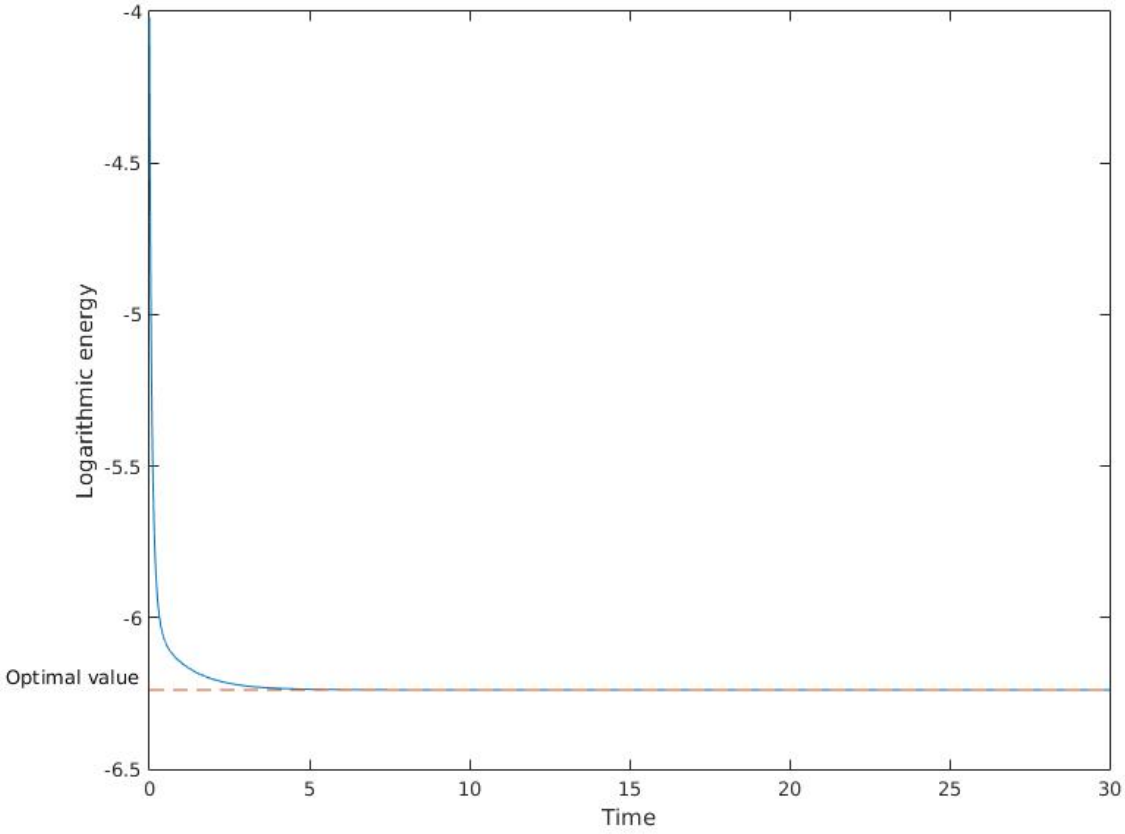

Figure 5.14: Logarithmic energy values for 6 Fekete spots over a time interval.

- Steady-state non-Fekete points

  We obtain a set of initial conditions are equilibria of the DAE system, but these equilibria are not elliptic Fekete points; Fig 5.15. The values of the logarithmic energy during the evolution of these spots are shown in Figure 5.16.

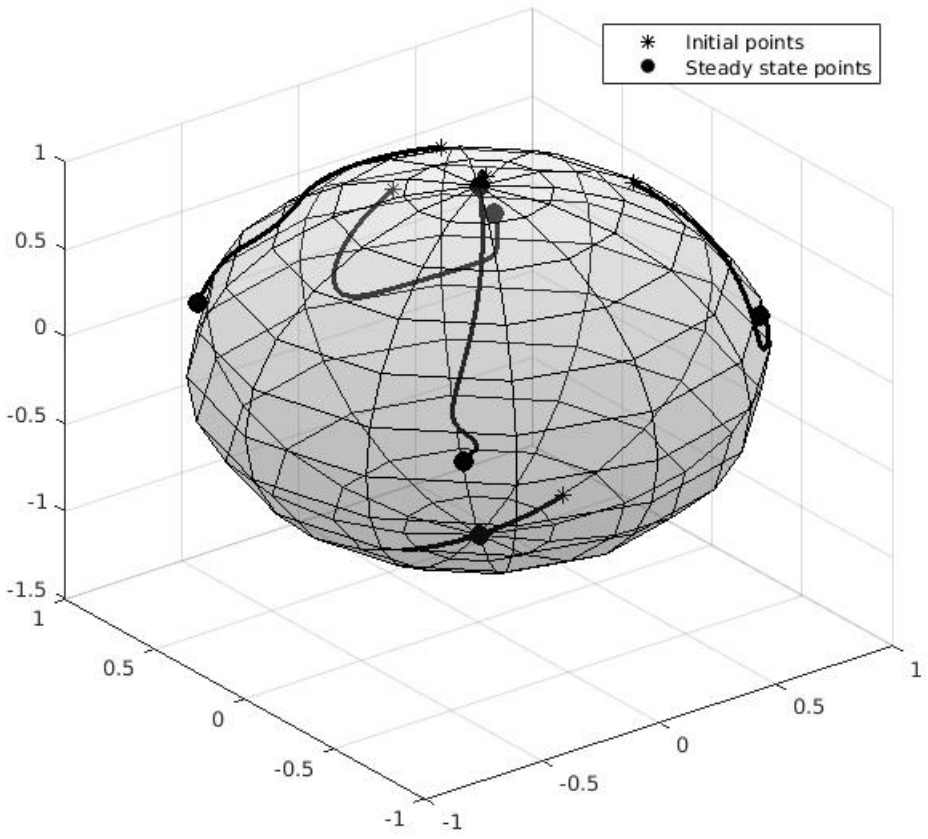

Figure 5.15: Non-Fekete Steady-State points

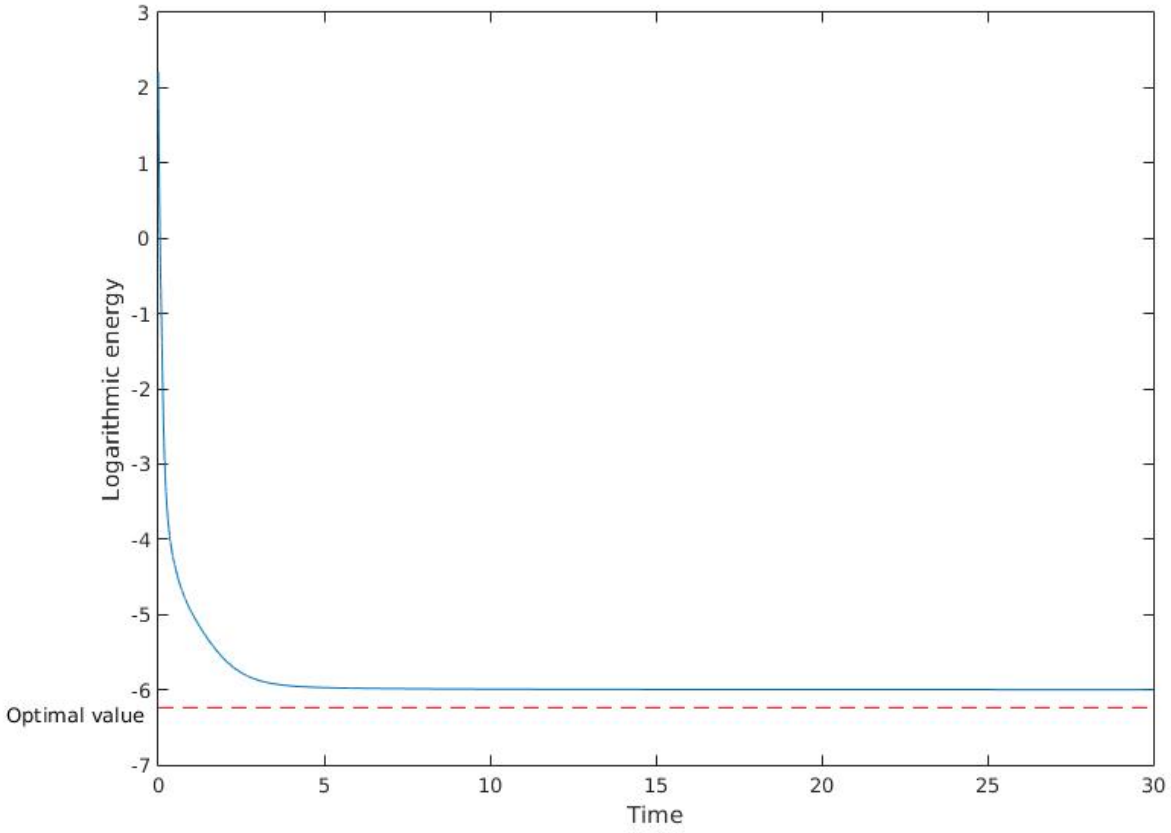

Figure 5.16: Logarithmic energy values for 6 non-Fekete spots over a time interval.

- Non-steady state

  Some initial conditions diverge; the numerical solutions become unbounded.

We remark that for $N \geq 9$, we were unable to find numerical solutions of the DAE system that converge to the Fekete points; numerical solutions either converged to non-Fekete points or became unbounded.

Chapter 6

Conclusions and Future Work

In this work, the dynamics of localized spot patterns on the unit sphere for the Brusselator RD model are studied. The DAE system that characterizes the slow dynamics of the spot pattern on long time-scale is presented. The connection between the equilibria of the DAE and the set of elliptic Fekete points is drawn.

6.1 Summary of the contributions

The contributions of this thesis can be summarized as follows:

First, the DAE system (3.9) is solved numerically using the Matlab's `ode15s` function. Section 3.4 shows the process of obtaining the numerical solutions. The spot patterns for $N = 6$ spots are presented.

Second, the equilibrium spot patterns are obtained by solving the Fekete optimization problem (5.1), the GBPSO method is used to calculate the minimal values of the logarithmic energy and hence the elliptic Fekete spots. The optimization computations are performed with `pyswarm` python software. The localized patterns for $N = 2, 3, \ldots, 12$ spots are presented.

Third, the connection between the equilibria of the DAE system and the elliptic Fekete points is studied in Chapter 5. It is shown for $N = 2, 3, \ldots, 8$ points, that it is possible to find conditions for the DAE system that converges at steady state to elliptic Fekete points. The connection for $N > 8$ could not be drawn.

6.2 Future work

Similar numerical, asymptotic, and optimization analysis can be extended for other RD models, such as the coupled bulk-surface model [31, 32] and the Gray–Scott Model [9], which model many biological and chemical phenomena, e.g., eukaryotic cell migration, chemotaxis, and signalling networks [23, 38]. The dynamics and equilibria of spot patterns for the Brusselator model on a general manifold, which may better model real surfaces, can be investigated. Group theory methods for ODEs can also be used to classify and identify the steady-state spot configurations of the DAE system; they can help in establishing numerical algorithm to classify the equilibria into symmetry groups, especially when $N > 8$ spots. Additional stability analysis of steady-state solutions of the DAE and core problem can be performed, providing a theoretical framework for the stability of spot patterns may help in performing efficient numerical computations.

References

[1] A. Lee. Particle swarm optimization (pso) with constraint support. https://github.com/tisimst/pyswarm/.

[2] W. Al-Hassan, M. Fayek, and S. Shaheen. PSOSA: An Optimized Particle Swarm Technique for Solving the Urban Planning Problem. In *Computer Engineering and Systems, The 2006 International Conference on*, pages 401–405, Nov 2006.

[3] M. S. Arumugam and M. V. C. Rao. On the performance of the particle swarm optimization algorithm with various inertia weight variants for computing optimal control of a class of hybrid systems. *Discrete Dynamics in Nature and Society*, 2006:Article ID 79295, 17 pages, 2006.

[4] U. M. Ascher and L. R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1998.

[5] B. Bergersen, D. Boal, and P. Palffy-Muhoray. Equilibrium configurations of particles on a sphere: the case of logarithmic interactions. *Journal of Physics A: Mathematical and General*, 27(7):2579, 1994.

[6] K. E. Brenan, S. L. Campbell, and L. R. Petzold. *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, volume 14. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1996. Revised and corrected reprint of the 1989 original.

[7] X. Cai, Z. Cui, J. Zeng, and Y. Tan. Self-learning particle swarm optimization based on environmental feedback. In *Innovative Computing, Information and Control, 2007. ICICIC '07. Second International Conference on*, pages 570–570, Sept 2007.

[8] M. Chaplain, M. Ganesh, and I. Graham. Spatio-temporal pattern formation on spherical surfaces: Numerical simulation and application to solid tumour growth. *J. Math. Biology*, 42:387–423.

[9] W. Chen and M. J. Ward. The stability and dynamics of localized spot patterns in the two-dimensional gray–scott model. *SIAM Journal on Applied Dynamical Systems*, 10(2):582–666, 2011.

[10] A. V. Christoph Landsberg. A multigrid finite element method for reaction-diffusion systems on surfaces. *Computing and Visualization in Science*, 13(4):177–185, apr 2010.

[11] S. Das, A. Abraham, and A. Konar. Particle swarm optimization and differential evolution algorithms: Technical analysis, applications and hybridization perspectives. In Y. Liu, A. Sun, H. Loh, W. Lu, and E.-P. Lim, editors, *Advances of Computational Intelligence in Industrial Systems*, volume 116 of *Studies in Computational Intelligence*, pages 1–38. Springer Berlin Heidelberg, 2008.

[12] L. C. W. Dixon and G. P. Szego. The global optimization problem: an introduction. *Towards global optimization*, 2(1-15), 1978.

[13] R. Eberhart and Y. Shi. Comparison between genetic algorithms and particle swarm optimization. In V. Porto, N. Saravanan, D. Waagen, and A. Eiben, editors, *Evolutionary Programming VII*, volume 1447 of *Lecture Notes in Computer Science*, pages 611–616. Springer Berlin Heidelberg, 1998.

[14] R. Eberhart and Y. Shi. Tracking and optimizing dynamic systems with particle swarms. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, volume 1, pages 94–100 vol. 1, 2001.

[15] A. P. Engelbrecht. *Computational Intelligence: An Introduction*. Wiley, second edition, 2007.

[16] EOS Guide. Types of optimization problems. http://www.neos-guide.org/optimization-tree.

[17] M. Fekete. Über die verteilung der wurzeln bei gewissen algebraischen gleichungen mit ganzzahligen koeffizienten. *Mathematische Zeitschrift*, 17(1):228–249, jan 1923.

[18] Frontline Systems Inc. Optimization problem types. http://www.solver.com/problem-types.

[19] Z. L. Gaing. Particle swarm optimization to solving the economic dispatch considering the generator constraints. *Power Systems, IEEE Transactions on*, 18(3):1187–1195, Aug 2003.

[20] Y. Gao, X. An, and J. Liu. A particle swarm optimization algorithm with logarithm decreasing inertia weight and chaos mutation. In *Computational Intelligence and Security, 2008. CIS '08. International Conference on*, volume 1, pages 61–65, Dec 2008.

[21] J. Gjorgjieva and J. Jacobsen. Turing patterns on growing spheres: The exponential case. *DCDS supplement (special issue),*, pages . 436–445, 2007.

[22] X. Hu and R. Eberhart. Adaptive particle swarm optimization: detection and response to dynamic systems. In *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, volume 2, pages 1666–1670, 2002.

[23] R. H. Insall and L. M. Machesky. Actin dynamics at the leading edge: From simple machinery to complex networks. *Developmental Cell*, 17(3):310 – 322, 2009.

[24] J. Kennedy. The particle swarm: Social adaptation of knowledge. In *Evolutionary Computation*, pages 303 – 308. IEEE International Conference on, 1997.

[25] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Neural Networks*, volume 4, pages 1942–1948. IEEE International Conference on, Nov 1995.

[26] J. Kennedy and R. Eberhart. A discrete binary version of the particle swarm algorithm. In *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation, 1997 IEEE International Conference on*, volume 5, pages 4104–4108, Oct 1997.

[27] T. Kolokolnikov, M. J. Ward, and J. Wei. The existence and stability of spike equilibria in the one-dimensional Gray–Scott model on a finite domain. *Applied Mathematics Letters*, 18(8):951 – 956, 2005.

[28] T. Kolokolnikov, M. J. Ward, and J. Wei. The stability of steady-state hot-spot patterns for a reaction-diffusion model of urban crime. *Discrete and Continuous Dynamical Systems - Series B*, 19(5):1373–1410, 2014.

[29] K. Lee and J. Park. Application of particle swarm optimization to economic dispatch problem: Advantages and disadvantages. In *Power Systems Conference and Exposition, 2006. PSCE '06. 2006 IEEE PES*, pages 188–192, Oct 2006.

[30] K. J. Lee, W. D. McCormick, Q. Ouyang, and H. L. Swinney. Pattern formation by interacting chemical fronts. *Science*, 261(5118):192–194, 1993.

[31] C. B. Macdonald, B. Merriman, and S. J. Ruuth. Simple computation of reaction–diffusion processes on point clouds. *Proceedings of the National Academy of Sciences*, 110(23):9209–9214, 2013.

[32] A. Madzvamuse, A. H. W. Chung, and C. Venkataraman. Stability analysis and simulations of coupled bulk-surface reaction–diffusion systems. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 471(2175), 2015.

[33] W. S. Nagata W, Lionel G. Harrison. Reaction-diffusion models of growing plant tips: bifurcations on hemispheres. *Bull Math Biol*, 65(4):571–607, 2003.

[34] R. Poli. Analysis of the publications on the applications of particle swarm optimisation. *J. Artif. Evol. App.*, 2008:4:1–4:10, Jan. 2008.

[35] I. Prigogine and R. Lefever. Symmetry breaking instabilities in dissipative systems. ii. *The Journal of Chemical Physics*, 48:1695–1700, 1968.

[36] A. M. R. Barreira, C. M. Elliott. The surface finite element method for pattern formation on evolving biological surfaces. *Mathematical Biology*, 63(6):1095–1119, Dec 2011.

[37] A. Ratnaweera, S. Halgamuge, and H. Watson. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *Evolutionary Computation, IEEE Transactions on*, 8(3):240–255, June 2004.

[38] A. Rätz and M. Röger. Turing instabilities in a mathematical model for signaling networks. *Journal of Mathematical Biology*, 65(6):1215–1244, 2011.

[39] I. Rozada, S. J. Ruuth, and M. J. Ward. The stability of localized spot patterns for the Brusselator on the sphere. *SIAM J. Appl. Dyn. Syst.*, 13(1):564–627, 2014.

[40] C. P. Schenk, M. Or-Guil, M. Bode, and H.-G. Purwins. Interacting pulses in three-component reaction-diffusion systems on two-dimensional domains. *Phys. Rev. Lett.*, 78:3781–3784, May 1997.

[41] T. Schoene. Step-optimized particle swarm optimization. Master's thesis, Department of Computer Science, University of Saskatchewan, Aug 2011.

[42] L. F. Shampine and M. W. Reichelt. The matlab ODE suite. *SIAM Journal on Scientific Computing*, 18:1–22, 1997.

[43] Y. Shi and R. Eberhart. A modified particle swarm optimizer. In *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, pages 69–73, May 1998.

[44] M. Shub and S. Smale. Complexity of Bezout's theorem: III. Condition number and packing. *Journal of Complexity*, 9(1):4–14, 1993.

[45] W. J. Stortelder, J. J. de Swart, and J. D. Pintér. Finding elliptic Fekete points sets: two numerical solution approaches. *Journal of Computational and Applied Mathematics*, 130(1–2):205 – 216, 2001.

[46] M. Taherkhani and R. Safabakhsh. A novel stability-based adaptive inertia weight for particle swarm optimization. *Applied Soft Computing*, 38:281 – 295, 2016.

[47] I. C. Trelea. The particle swarm optimization algorithm: convergence analysis and parameter selection. *Information Processing Letters*, 85(6):317 – 325, 2003.

[48] P. H. Trinh and M. J. Ward. The dynamics of localized spot patterns for reaction-diffusion systems on the sphere. *ArXiv e-prints*, Nov. 2014.

[49] M. Tsuji. *Potential theory in modern function theory.* Chelsea Pub. Co, second edition, 1975.

[50] F. van den Bergh. *An Analysis of Particle Swarm Optimizers.* PhD thesis, University of Pretoria, Faculty of Natural and Agricultural Science, 2001.

[51] V. K. Vanag and I. R. Epstein. Localized patterns in reaction-diffusion systems. *Chaos*, 17(3), 2007.

[52] C. Varea, J. L. Aragón, and R. A. Barrio. Turing patterns on a sphere. *Phys. Rev. E*, 60:4588–4592, Oct 1999.

[53] M. Zambrano-Bigiarini, M. Clerc, and R. Rojas. Standard particle swarm optimisation 2011 at cec-2013: A baseline for future pso improvements. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 2337–2344, June 2013.